Spring 2017

Mapping Data Visualization to Timbral Sonification and Machine Listening

Dale E. Parson
Kutztown University, parson@kutztown.edu

Wissam Malke
Kutztown University of Pennsylvania, wmalk292@live.kutztown.edu

Halley Langley
Kutztown University of Pennsylvania, hlang151@live.kutztown.edu

Danielle Emily Hoch
Kutztown University of Pennsylvania, dhoch285@live.kutztown.edu

MAPPING DATA VISUALIZATION TO TIMBRAL SONIFICATION AND MACHINE LISTENING

Dale E. Parson, Wissam Malke, Hallie Langley, and Danielle Emily Hoch

Kutztown University of PA,
Department of Computer Science and Information Technology,
P.O. Box 730, Kutztown, PA, USA, 19530
parson@kutztown.edu

ABSTRACT

Parallel coordinate plotting is a data visualization technique that provides means for exploring multidimensional relational datasets on a two-dimensional display. Each vertical axis represents the range of values for one attribute, and each data tuple appears as a connected path traveling left-to-right across the plot, connecting attribute values for that tuple on the vertical axes. Parallel coordinate plots look like time-domain audio signal waveforms. This study investigates several timbral data sonification algorithms for classification in which audio waveforms derive from the shapes of parallel coordinate tuple plots of data being classified. Listening-response survey results and analyses reveal that mapping parallel coordinates of data tuples to audio waveforms can be accurate for generating sounds that human listeners can use to classify data. This study also investigates using machine learning algorithms to build a machine listener that approximates human survey taker performance in classifying data sounds.

Keywords: data sonification, data visualization, machine listening

1. INTRODUCTION

This report presents an investigation into the relative effectiveness of several variations of a technique for the perception-based classification of individual data records (a.k.a. instances or tuples) in a relational dataset by using sonification of attribute values for each attribute in a record. Sonification is the process of mapping data attribute values to properties of sound [1]. Sonification is the aural counterpart to visualization, which is the process of mapping data attribute values to visual structures, for example in computer graphical displays. Sonification and visualization play roles in at least two stages of data analysis [2]. They serve the mechanisms of perceptual pattern recognition, helping the respective auditory and visual cognitive systems of an analyst to detect patterns in data as a guide to subsequent formal analysis. They can also serve to illustrate relationships found through formal analysis, coming after formal analysis. The current investigation relates to the former role, the use of perceptual pattern matching in exploring and classifying data instances. This paper reports the current stage of a research track of using synchronized sonification and visualization for exploring datasets.

Because of familiarity, the authors sonified attributes of a dataset that is part of an ongoing study into the correlation of temporal work habits of computer programming students to their success in projects as measured by project grades [3,4]. The authors anticipated the fact that familiarity with the dataset would make it easier to detect sonification opportunities and mistakes. However, the approaches explained in this report are domain neutral. They can apply to any relational dataset in which some attributes co-vary according to instance membership in disjoint sets of instances to be classified.

The format of the first year of the study consisted of conducting and analyzing so-called *sonic surveys* of sonified datasets, in which human listeners classify the sounds of individual data records by matching each to the closest of one of three reference sounds, where a *reference sound* is a statistical aggregate of all records in a given class. The second year replaced the human survey takers with machine learning algorithms that classify the audio (WAV) files used in the human sonic surveys in attempts to match human performance, partly in the interest of automating evaluation of new sonification algorithms. The goal of the sonification research is to find good sonification approaches as perceived by humans. Human-like performance of a machine listener provides a means for estimating the distinctiveness of a sonification algorithm as perceived by humans, so that when we eventually run sonic surveys again, we can try out the best sonification algorithms as suggested by the machine listeners. We hope to find a high-accuracy sonification algorithm as perceived by humans, and we also hope to verify the soundness of the machine listener. This latter part of the study has led to new results in machine listening with very small training datasets. This paper reports new results from the second half of year one, along with initial results in machine listening and its application to evaluating additional sonification algorithms.

2. RELATED WORK

The primary influence on the attribute sonification techniques of this report is the visualization technique of using parallel coordinates [5]. Figure 1 is a typically dense parallel coordinates plot of 22 of the 106 attributes found in the student work habit dataset of 282 student-project records [3,4]. Each vertical line represents the overall range of one attribute, with the minimum value at the numeric label near the bottom, and the maximum value at the top;

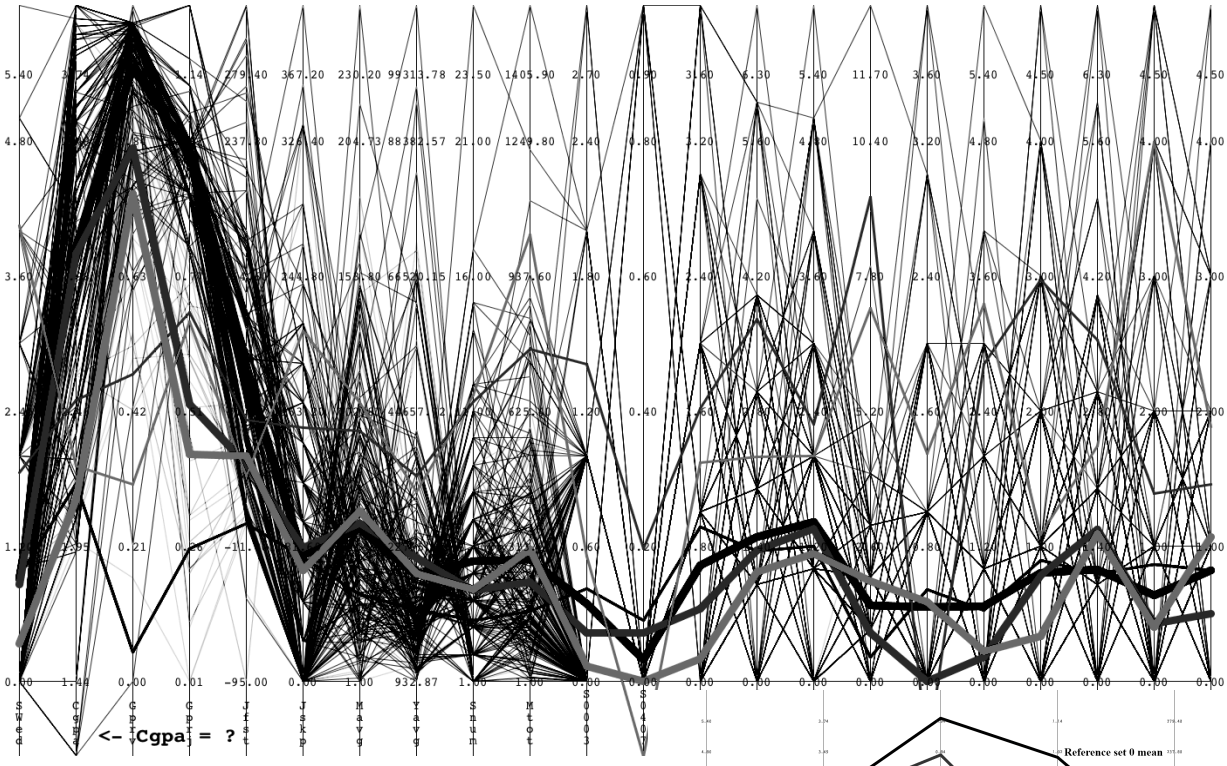

Figure 1: Parallel Coordinates Plot of 22 of the 106 attrib
The thick paths show mean values for three distinct sets of

an unknown value is at the very bottom. Figure 1's second vertical axis from the left, for example, bears the label "Cgpa" for "computer science grade point average", ranging from 1.44 near the bottom to 4.0 at the top. Going left to right, each thin multi-segment path represents one record in the dataset, intersecting a vertical axis at the point of the value for that record's attribute. The reduction of 106 attributes to the 22 in Figure 1 was part of a reduction of the scope of exploration of this dataset determined by the previous studies [3,4].

The homegrown software tool used to create Figure 1 uses partial transparency to plot the 282 instances in this dataset so that their paths do not obscure each other. Overlapping path segments increase opacity, which appears as brightness on a color computer screen. Figure 1 also shows three thick paths and three mid-thickness paths for the mean and population standard deviation, respectively, of three sets of instances. The thick black path shows the mean of all attributes for records with a mean project grade (Gprj, the fourth attribute from the left) that is >= 80%; we refer to this set of records as Reference Set 0. The thick medium-gray path shows the mean of all attributes for records (Set 1) with a mean project grade < 80% and a computer science grade point average (Cgpa) >= 2.5. The thick light-gray path shows the mean of all attributes for records (Set 2) with a mean project grade < 80% and a computer science grade point average < 2.5. Projects in this course serve more as learning exercises than as tests, so a grade that is < 80% is a poor grade. The mean project grade for all records is 92.3% with a standard deviation of 21%. The maximum project grade for one project per semester is 125% because of bonus points, giving the top of the Gprj range in Figure 1.

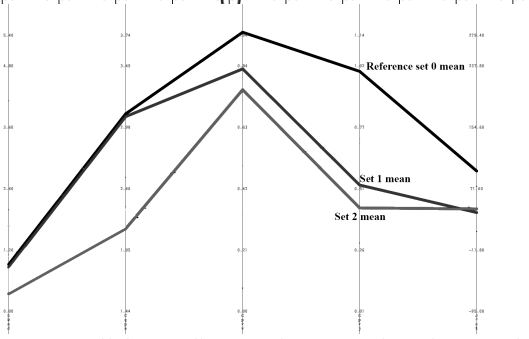

Figure 2: Parallel Coordinates of Mean Values for SWed, Cgpa, Gprv, Gprj & Jfst Attributes by Set

A less cluttered parallel coordinates plot containing the mean values for the leftmost 5 attributes of Figure 1 appears in Figure 2. Informally, Reference Set 0 is the set of all student-projects with high grades, Set 1 is the set with low grades and high Cgpa values, and Set 2 is the set with low grades and low Cgpa values. Sets 1 and 2 represent at-risk students who are at risk for potentially different reasons. Figure 2 includes set identifier labels.

An early observation was that the multi-segment paths of Figures 1 and 2 look a lot like audio time-domain waveform plots. A triangle wave can be modeled as the sum of a fundamental sinusoidal waveform and a series of its positively weighted odd harmonics [6], where a harmonic is a frequency multiple of the fundamental. A sawtooth wave can be modeled as the sum of a fundamental sinusoidal waveform, its positively weighted odd harmonics, and its negatively weighted even harmonics. These particular sounds become significant later in this discussion. The main point for now is the similarity in the shapes of multi-segment record paths in the parallel coordinates plots of Figure 1 and 2 on the one hand, and time-domain audio waveform plots on the other. That similarity, and the potential isomorphism between parallel coordinate plots and audio waveforms that yield distinct timbres, provide the inspiration and basis for this research.

Neuhoff presents a taxonomy of applying pitch, loudness, and timbre as the primary approaches with which to sonify data [7]. Duration of sound, spatial location, and sequences of distinct sounds are additional approaches.

Recent work that has inspired the present study involves the sonification of material x-ray scattering data by mapping two-dimensional arrays of x-ray intensity values directly to two-dimensional arrays of sound frequency components that define an audio waveform (timbre) [8]. That approach is similar to the approach of the present study in mapping domain data directly to waveforms (timbre) while avoiding any kind of musical or other aesthetic interpretation of the data that might introduce arbitrary sonic artifacts. In contrast to that work, the hallmark of the present study is the use of parallel coordinates plots of domain data as the source of mappings to sound.

Our previous paper reports on three substantially different approaches to sonifying this dataset and similar relational data [9]. The *harmonic* and *melodic* approaches convert attribute values to pitch rather than timbre, with the former sounding attribute-derived harmonic intervals simultaneously, and the latter sounding the same pitches as melodic intervals in temporal sequence. The third, timbral approach, *waveform sonification*, converts parallel attribute graphs such as Figure 2 directly into time-domain audio waveforms. This waveform approach resulted in the highest number of accurate sound classification responses among human survey takers. Because of its accuracy, the work presented here focuses on improving waveform sonification.

3. CLASSIFICATION THROUGH SONIFICATION

The present study uses sound for instance classification. The modus operandi is to investigate competing approaches for sonifying the dataset summarized in Figures 1 and 2. Our approach generates a reference sound for each of the three sets of means of Figure 2, and it generates a sound for each data record contributing to one of those means. A listener classifies a data record's sound as being closest to Set 0's reference sound, or Set 1's sound, or Set 2's, thereby classifying the record as belonging to Set 0, 1, or 2. The experiments reported include competing sonification methods for turning the mean reference records and individual data records into sounds.

3.1. Reducing the number of attributes to sonify

There are a total of 282 student project records in the dataset, where each record shows the work patterns (primarily temporal patterns) and performance of one student completing one programming project. With 106 attributes per record, there is a total of 282 x 106 = 29,892 data points. After eliminating redundant attributes, non-continuous attributes, and non-useful attributes as determined by the previous studies [3,4,9], the 22 attributes of Figure 1 remain. Semi-automated classification tools within the Weka data analysis toolset [10,11] provide one means for reducing the 22 attributes of Figure 1 down to the 5 attributes for sonification of Figure 2. The CfsSubsetEval attribute evaluator of Weka evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them in predicting the value of a target attribute such as project grade. This attribute evaluator indicates that three of the first five attributes appearing in Figure 1, namely computer science grade point average coming into the course (Cgpa), the grade on the previous project (Gprv), and the number of hours a student started a project before its deadline MINUS 24 hours for each day the student did not work on the project (Jfst) are the three best indicators for a record's project grade (Gprj).

The other, interactive, visual means for reducing the 22 attributes of Figure 1 down to the 5 attributes of Figure 2 comes from using our homegrown software tool for interacting with parallel coordinates data displays to find attributes with diverging means. Recall from the previous section that Reference Set 0 of Figures 1 and 2 consists of student-project records with a project grade (Gprj) that is >= 80%. Set 1 consists of student-project records with a project grade (Gprj) that is < 80% and a computer science grade point average (Cgpa) that is >= 2.5, and Set 2 consists of student-project records with a project grade (Gprj) that is < 80% and a computer science grade point average (Cgpa) that is < 2.5. Diverging mean values for multiple attributes in Figures 1 and 2 provide a basis for attribute-distance-based sonification. Our experimental approaches sonify the distance between a given record's attribute value and the Set_0_mean for that attribute, as a function of the Set_0_standard deviation for that attribute. An attribute value within 1 Set 0 standard deviation of the Set 0 mean generates a sound property that is relatively sweet; an attribute value within 2 standard deviations generates a sound that is a mix of sweet and sour; and an attribute value greater than 2 standard deviations generates a sound that is all sour. The next subsection discussion quantifies "sweet" and "sour", and explains their application in generating sounds.

A final concern is distinguishing Set 1 instances from Set 2 instances. All attributes in Figure 2 except Jfst (lead time MINUS 24 hours for unworked days) have distinct mean values for Sets 1 and 2. The Cgpa attribute is the defining difference between these two sets. Therefore, a good sonification algorithm should have sufficient data for generating sounds that distinguish Set 1 from Set 2 instances. We limited the number of attributes to 5, based on our Weka analysis and inspection of the parallel coordinates, to limit the complexity of the sounds. Too few attributes do not distinguish set membership of individual data records adequately, while too many generate complicated and confusing sounds.**Required parameters are missing or incorrect.**

3.2. Sweet and sour sonic boundaries

The sonification algorithms in this study use a helper algorithm that we call *sweetAndSour* to extract two numeric values for each attribute in either a Set of mean values or in an individual data record. This algorithm first computes the difference between an attribute being sonified and that attribute value for Reference Set 0, which is the mean of the reference set of records as defined above. If the attribute being sonified lies within 1 population standard deviation of Reference Set 0 for that attribute, it receives a sweetWeight in the range [0.33, 1.0]; the 1.0 end of that continuum is for a value that equals the Reference Set 0 value, and the 0.33 is for one standard deviation away; and it receives a sourWeight of 0.0. If the attribute being sonified lies within the range (1, 2] standard deviations of

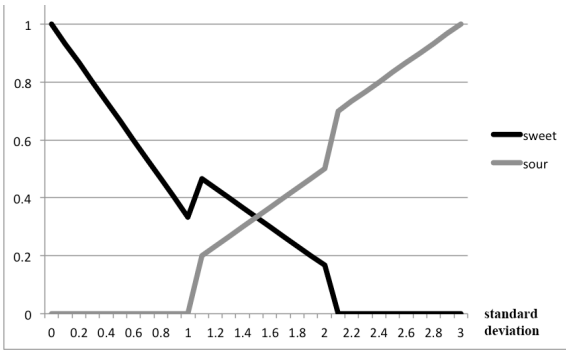

Figure 3: Sweet and Sour values as a function of attribute standard deviation

Reference Set 0 for that attribute, it receives a sweetWeight in the range [.167, .5) and a sourWeight in the range (.167, .5], with the maximum standard deviation of 2.0 giving a sweetWeight of .167 and a sourWeight of .5. The further the distance from the Reference attribute mean, the less sweet and more sour the sweetAndSour numbers. Finally, the algorithm clamps the standard deviation at 3.0. If the attribute being sonified lies within the clamped range (2, 3] standard deviations of Reference Set 0 for that attribute, it receives a sweetWeight of 0.0 and a sourWeight in the range (.667, 1.0]. Figure 3 shows the curves for these values. The temporary change in direction of the sweet curve at the standard deviation of 1.0 was an unintentional bug, but by the time it was discovered, experimental classification response collection had already begun. The mix of sweet and sour differs to the immediate left versus right of the stddev = 1.0 point, and furthermore, the overall non-linear, step-function relationship of the curves was intentional. Each attribute contributes some amount of sweet versus sour, each in the range [0.0, 1.0], and within a standard deviation band, these parameters vary linearly. The intent for this approach is to cause abrupt changes in sound when crossing a discrete boundary. We fixed the bug and created a variant of this sweet & sour graph without the kink after completing all human survey data collection. Section 5 on future work discusses nonlinear alternatives to this sweet & sour approach.

3.3. Waveform sonification

The *waveform sonification* algorithm derives from the observation that the parallel coordinate plots of Figures 1 and 2 look like waveforms. In fact, the first 5 attributes of Figure 1, which are the attributes of Figure 2, are sorted to approximate a triangular waveform for the Reference Set 0 mean values. The original idea was, to the degree that Set 1 and 2 waveforms deviate from the Set 0 waveform, a listener would distinguish timbral differences and classify into the three sets on the basis of those.

Mapping the parallel coordinates plots directly to audio waveforms created sounds that were hard to distinguish, so we came up with the idea of using sweet and sour parameters to introduce discontinuities. For each attribute of a record, after determining the sweet and sour parameters, the waveform generator tests which is greater in magnitude, sweet or sour. For attributes where sweet dominates, it saves the attribute value in its position. For attributes where sour dominates, it saves the additive inverse (the "negative") of the attribute value in its position. The intent

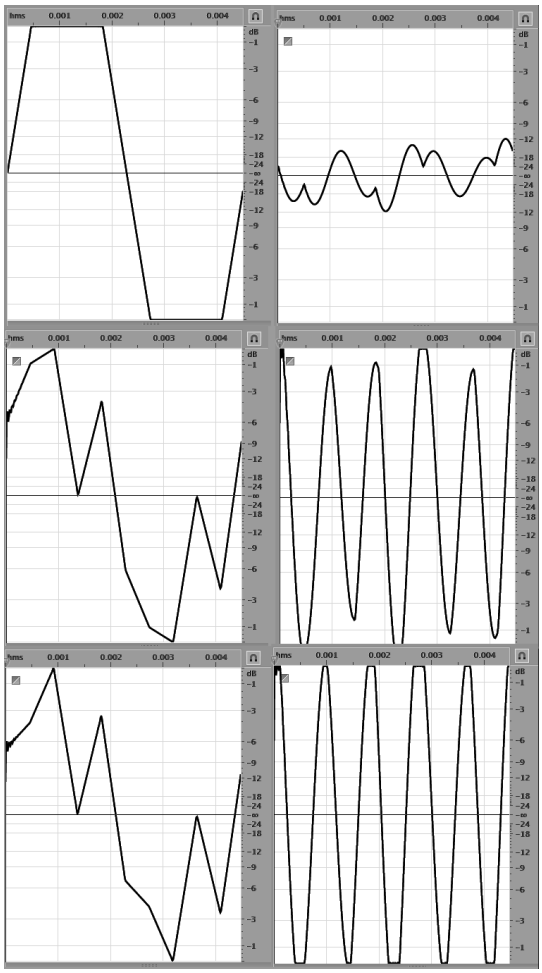

Figure 4: Unfiltered (left) and filtered waveforms for Reference Set 0, Set 1, and Set 2 mean values for SWed, Cgpa, Gprv, Gprj & Jfst Attributes

is to create more "kinks" in a sour attribute's inflection point in a waveform, increasing overtone frequencies (partials). After traversing all attributes of a record, the waveform generator normalizes the range of values to the range [0.0, 1.0] by scaling. It then generates a ChucK program [12] that plays this waveform for 2 seconds. The sounded waveform is actually the original waveform in the [0.0, 1.0] range, and then its mirror image in the [-1.0, 0.0] range, in order to preserve symmetry and avoid introducing additional overtones.

The left side of Figure 4 shows the original waveforms for Reference Set 0, Set 1, and Set 2, starting at the top. Reference Set 0 is somewhat problematic because all of its attributes are 100% sweet and 0% sour, with zero difference from the mean, giving a flat line. The waveform sonification algorithm treats the lowest attribute value as a minimum and scales the others according to their range, winding up with a near square wave for Set 0 at the top left of Figure 4.

Set 1 and 2 original waveforms appear below Reference Set 0. Inspection shows 5 vertices in the positive, initial side of the Set 1 waveform, corresponding to the five attributes SWed, Cgpa, Gprv, Gprj and Jfst. There is the initial point (SWed), a slight bend to a lesser slope near .45 ms. (milliseconds) (Cgpa), a peak (Gprv), a trough at the 0 center line (Gprj), and a final peak (Jfst) before going to the negative mirror half of the waveform. The overall

waveform occupies 4.5 ms., which is 1.0 / 220 Hz baseline frequency. The trough for the second-last attribute (Gprj) in the Set 1 and 2 waveforms corresponds to the distances between their means and the Reference Set 0 mean in Figure 2. Gprj is the only parameter for which the deviation of Sets 1 and 2 are so great that they generate a negative-going trough, which contributes overtones in the timbre. Cgpa, the attribute for which Sets 1 and 2 differ from each other most significantly, gives a reduction in slope for Set 1, and an increase in slope for Set 2, in going from Cgpa to Gprv.

The original sounds from the left side of Figure 4 are so dominated by 220 Hz and other low-frequency harmonics that it is hard to distinguish them when listening. Since the intent was to generate high-frequency markers in the non-reference Sets 1 and 2, we added a high-pass filter operating at 4.5 X the baseline frequency of 220 Hz, with a filter factor that makes it moderately selective (filter Q = 10). It passes frequencies above 990 Hz with relatively little attenuation, and it allows some frequencies below that threshold to pass with gentle but increasing attenuation as frequency goes down. We picked these values for the filter through listening. The waveforms on the right side of Figure 4 are the results of high-pass filtering. Reference Set 0 at the top has its amplitude diminished considerably because it consists mostly of low-frequency components that manage to make it past the filter. Sets 1 and 2 show more remaining amplitude because of their sour-parameter-generated overtones, with Set 2 saturating at the -1.0 and 1.0 limits in more places than the Set 1 waveform. The waveforms on the right side of Figure 4 are the ones actually used in the surveys of the next section.

3.4. Conducting and analyzing the sonic surveys

In addition to generating sounds, running the ChucK programs generates uncompressed WAV (Waveform Audio File Format) files that store those sounds. For the sonic surveys we created a Java survey application that loads the mean reference set and individual data record WAV files and presents them to listeners via a GUI and desk monitors, adjusted by one of the authors to safe levels. The sonic survey allows the three Set reference tones for a given sonification algorithm to be sounded any time while manually sequencing through 39 pseudo-randomly selected data record sounds, 13 belonging to each of Set 0, 1, and 2. The listener selects the Set 0, 1, or 2 that they feel is closest in sound to current data record sound, and then goes on to the next record. The numbers of sounds are a function of the numbers of records in the least-populated set of data. After making responses to each of the 39 sounds in the first sonification approach, a listener listens to three reference set sounds and then responds to 39 instances for a second sonification algorithm, and then listens to three final reference set sounds and then responds to 39 instances for a third sonification algorithm. Survey completion takes about 20 minutes. There were 29 volunteer participants in the fall 2015 survey and 33 in the spring 2016 survey. There was no data collection about familiarity with music or computer audio, and no discussion about the data or sonification techniques used in the sonic survey. There were 117 selection mouse clicks (39 sounds for each of 3 sonification approaches) X 29 listeners = 3393 data points, 1131 per sonification technique for the fall, and 117 X 33 = 3861 data points, 1287 per sonification technique for the following spring. The order of presentation of sonification algorithms and sonified data records was randomized for each person. Survey takers showed no signs of performance benefit for the first algorithm presented, which might be due to lack of listening fatigue, or for the last, which might be due to learning. Order of algorithm presentation showed no statistical significance.

Table 1 shows the mean values of correct sonic record classifications by survey takers, and the sample standard deviations, for each sonification algorithm across all sets (classes) and for each of the three sets. *Waveform* is the only algorithm presented here from the fall 2015 surveys. It uses the sonification algorithm detailed in section 3.3 and illustrated on the right side of Figure 4, as first reported in [9]. The other waveform variants are from the spring 2016 surveys, first reported here. *WaveX2* makes two identical copies of the initial, 220Hz-fundamental Waveform, doubles the frequency of one copy, thereby moving it up an octave, and mixes the two waveforms in the ChucK generator program, normalizing the amplitude of the summed waves. *WaveX4/3* works similarly, multiplying the frequency of one copy by 4/3 (a just fourth) instead of 2, and *WaveX1.95* multiplies the frequency of one copy by 1.95. The idea is to investigate two forms of audio consonance with WaveX2 and WaveX4/3, and dissonance with WaveX1.95.

Looking first at "All 3 sets" entries, which give the overall results for each sonification algorithm, WaveX2 is the most successful, with the highest mean correct response rate of 67.8%, and the lowest sample standard deviation of 13.5. Note that a random guess from 1 out of 3 classes would give an expected mean result of 33.3%. WaveX2 more than doubles this number, but it is far from perfect.

For Set 0, Waveform is the marginal winner at 74.8% mean correct responses, with WaveX2 and WaveX1.95 tied at 73.7%. The slight loss in accuracy is not surprising, given the fact that the two spring 2016 algorithms generate more overtones than the basic Waveform approach. Paucity of overtones is a hallmark of Set 0 Waveform sonification.

WaveX2 ties Waveform for Set 1 at 57.8%, and WaveX4/3 has the best result for Set 2 at 77.6%. WaveX2 is the overall "winner" because of its "All 3 sets" performance and its lack of distinctly sub-par results for

Sonification algorithm	Category	Mean correct responses	Sample stdev across survey takers
Waveform	All 3 sets	61.4%	17.4
Waveform	Set 0	**74.8%**	13.0
Waveform	Set 1	**57.8%**	12.8
Waveform	Set 2	51.5%	17.2
WaveX2	All 3 sets	**67.8%**	**13.5**
WaveX2	Set 0	73.7%	10.7
WaveX2	Set 1	**57.8%**	14.3
WaveX2	Set 2	71.8%	9.4
WaveX4/3	All 3 sets	65.8%	19.0
WaveX4/3	Set 0	72.3%	16.7
WaveX4/3	Set 1	47.6%	13.0
WaveX4/3	Set 2	**77.6%**	11.6
WaveX1.95	All 3 sets	67.6%	15.9
WaveX1.95	Set 0	73.7%	17.9
WaveX1.95	Set 1	57.6%	14.4
WaveX1.95	Set 2	71.6%	9.3

specific sets. The desire to investigate additional variants of Waveform sonification for further improvements, without the time and expense of conducting human surveys, led us to the machine listening approach to surveys explained in the next section. We defer examining spectral plots of the Set 0, 1, and 2 reference sounds to the following section.

4. MACHINE LISTENING WITH VERY SMALL TRAINING DATASETS

4.1. Approximating human survey response accuracy in a machine listener

We used the Weka data mining toolset [10,11] to classify the sounds described in the previous section because of our familiarity with Weka and because it comes with a large repertoire of machine learning algorithms. One limitation for Weka is that the data for training and testing datasets must fit into main memory. The memory load for 3 training instances + 39 test record instances for one *virtual survey* is small, but we needed to run many virtual surveys using many machine learning algorithms with many variations of their parameters. Given available data, we ran 50 virtual surveys of 3 training instances + 39 test record instances for 21 distinct variations of Waveform sonification, including the 3 spring 2016 WaveX variants of Table 1. In addition to these variants, we used two almost-identical variants, one of which removes the kink-bug of Figure 3 (so-called "fixed" sonification algorithms), and one of which removes the non-linear boundaries of Figure 3 altogether, drawing straight Sweet and Sour lines diagonally across the figure (so-called "linear" algorithms). That gives 9 related variants of WaveX. In addition, there are 6 waveform variants that generate sawtooth waveforms by sorting the Reference Set 0 attributes of Figure 2 in lowest-to-highest order (3 for "fixed" and 3 for "linear", using sums of two copies of the waveform), and there are 6 waveform variants that generate reverse-sawtooth waveforms by sorting the Reference Set 0 attributes of Figure 2 in highest-to-lowest order (3 for "fixed" and 3 for "linear"), totaling 21 distinct variations of WaveX sonification. We did not apply the Figure 3 kink-bug to the sawtooth variants when we found that the "fixed" Wave2X version outperforms the buggy version during an intermediate stage of work.

Data preparation for Weka consists of using a ChucK analysis program to extract, from each training (i.e., 3 reference sets) and test (39-instance records) WAV file, the following sonic parameters: a Fast Fourier Transform (FFT) histogram of frequency magnitude, using a 128 Hanning window size, yielding 64 distinct FFT values in each histogram bin, along with the WAV file Centroid (central frequency), root-mean-square (RMS) power, and the frequencies at which 25%, 50%, and 75% of the signal energy begins to attenuate (roll-off frequencies). Note that this use of ChucK is different from the previous use for generating WAV files from relational data. Here, ChucK "listens" to the same WAV files as the human survey takers, and writes measurements for these sonic parameters to a text file, which a Python script then formats for Weka's Attribute-value Relational File Format ("ARFF").

We tested a total of 12 distinct machine learning algorithms in Weka, based on initial manual investigation and subsequent iterative, automated testing [13]. We varied configuration parameters for all of these algorithms, giving a grand total of 163 machine learning algorithms with parameter variations. Exploring the configuration parameter space of these algorithms is necessary because of random aspects of some and high sensitivity to configuration parameter values of some. This gives an overall total of (3 training + 13 testing records) X 50 virtual surveys X 21 sonification algorithms X 163 machine learning variants = 2,738,400 distinct sonic record classifications. The goal in this large search was to find machine learning algorithms that approximate human survey taker accuracies, and then to explore additional sonification algorithms using these human-comparable machine learning algorithms. We constructed our Machine Learning Evaluator Tool in Java to run the sonic virtual surveys through Weka in batch, command-line mode.

The first discovery when inspecting classification results for Weka's machine learning algorithms that are typically powerful, such as the J48 decision tree builder [14], is that they achieve an average of only 33.3% correct classifications, which is to say random guessing of the best match of each sound record to references sounds 0, 1, and 2. These results are unsurprising for those classifiers because they require large training datasets in order to produce accurate models of data relationships. This problem highlights the fact that our human-oriented classification problem relies on an extremely small training set size of 3 records, a size that is miniscule when compared to modern, big-data classification problems. Human short-term memory for reference sounds is very limited in capacity. The problem is one of finding classifiers that work well with extremely small training datasets.

Weka's *RandomTree* decision tree classifier with options –K (number of randomly chosen attributes) = 4, -M (minimum weight of a leaf in the tree) = 1.0, and –S (random seed) = 0, is the only classifier that gives the same range as the human survey results, where the classifier result ranges are between 64% and 65%. RandomTree considers K randomly chosen sonic attributes at each tree node while building its model [11]. Chosen attributes significantly affect the accuracy of the model, in the sense that, the more dominant these attributes are, the more accurate the model. In addition, RandomTree classifier does not perform pruning; lack of pruning retains all nodes and paths in its classification trees. Retaining as much decision logic as possible is especially useful in the case of small training dataset. Pruning is more appropriate when there are large training sets that need to be generalized, in part, via pruning. This classifier is capable of producing a model with results equivalent to the human results using very small training datasets. The details of a small training dataset used to build a classifier are very crucial for its accuracy. 121 of the aforementioned 163 machine learning algorithms with parameter variations are variations of RandomTree. While tree construction builds strictly on the 3 training set instances and the configuration parameters, selection of the most-human-performing configuration parameters relies on testing with the (3 training + 13 testing records) X 50 virtual surveys X 121 RandomTree variants = 96,800 distinct sonic record classifications. Over-fitting is not a concern, given the fact that 96,800 records present a very large representation of the surveys taken by humans.

This most human-like-response RandomTree classifier is quite simple, with the following structure, where the 64

FFT histograms start at fft0, and a given fft strength scales against a normalized upper value of 1.0.

If Centroid < 1764 Hz (.08 * 22,050)

Classify as Set 0.

Else if fft6 centered at 2412 Hz < .03

 # .03 is 3.48 on Fig. 5 log scale.

Classify as Set 1.

Else

Classify as Set 2.

The top half of Figure 5 shows the first 11 bins of the 64-bin-Hanning FFT for the three Wave2X training WAV files, with fft6 appearing as a vertical bar centered at 2412 Hz., and the actual centroids as measured by ChucK tagged with a "C", e.g., "C0" for the frequency centroid for Reference Set 0. ChucK extracts the data of Figure 5 for Weka. Note that there is a small RandomTree error for Set 0's mean centroid of 1804 Hz., which exceeds the 1764 Hz. of the above generated decision tree. Small rounding errors in the RandomTree precision account for the difference. The height of each bar in Figure 5 is $\log_{10}$(the fractional bin value from ChucK) + 5, where the "+ 5" term compensates for negative log values of fractions. Perception of loudness follows a logarithmic scale [15]. For distinguishing Set 1 from Set 2 instances, there is correctly more total signal strength for Set 2 within the fft6 bin. These small signal differentials are enough to allow RandomTree to classify sonic records with the same accuracy as humans. We make no assertions about whether these are the attribute measures used by humans, but they are certainly viable.

The two Weka classifiers that achieved the best, extra-human accuracy for Wave2X are K* and AdaBoostM1 with the HoeffdingTree underlying learner, achieving classifier result ranges between 68% and 78%. *K** is an instance-based learner that does not derive a decision tree or other generalized, abstract model structure. The class of a test instance is based upon the class of those training instances similar to it, as determined by a similarity function. K* measures distance between test and training attributes according to an entropic information transform, i.e., the distance in terms of edits or normalized numeric delta to transform a test attribute into a training attribute. It differs from other instance-based learners in that it uses an information-entropy-based distance function. This process appears similar to the human learning and classification process of the surveys, in the sense that a human listens to three training instances, then uses those memories as reference points for classifying test instances based on perceived similarity. Humans always have access to training sounds during the surveys, making it possible to compare test sounds to training sounds on a case-by-case basis.

AdaBoostM1 is an ensemble learner that applies a base case learner multiple times in performing tests. For the 68% to 78% accuracy range for Wave2X, the base case classifier is *HoeffdingTree*. HoeffdingTree exploits the fact that a small sample can often be enough to find an optimal splitting attribute. This idea is supported mathematically by the Hoeffding bound, which quantifies the number of observations needed to estimate some statistics within a prescribed precision [16]. This classifier conforms to the principle with our main goal of using a small training dataset.

4.2. Evaluating new sonification algorithms via a machine listener

The final portion of this study consists of evaluating additional sonification algorithms, beyond those used in the survey of humans. As previously noted, there are 6 waveform variants that generate sawtooth waveforms by sorting the Reference Set 0 attributes of Figure 2 in lowest-to-highest order, 3 for "fixed" and 3 for "linear", using sums of two copies of the waveform; there are 3 for each

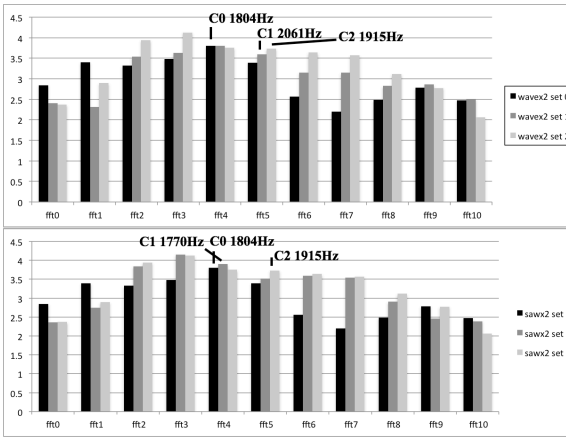

Figure 5: Spectral data for WaveX2 and SawX2 training WAV files; Y axis is a $\log_{10}$ scale of the FFT bin levels

because of the 2X, 4/3X, and 1.95X summed variations of Table 1, applied to sawtooth. Similarly, there are 6 waveform variants that generate reverse-sawtooth waveforms by sorting the Reference Set 0 attributes of Figure 2 in highest-to-lowest order, 3 for "fixed" and 3 for "linear". There are also 6 "fixed" and "linear" variants of the spring 2016 WaveX sonifiers of Table 1, for a total of 18. We applied Weka's RandomTree decision tree classifier that approximates human performance, with options previously discussed {–K (number of randomly chosen attributes) = 4, -M (minimum weight of a leaf in the tree) = 1.0, and –S (random seed) = 0}, to this collection of 18 sonification algorithms. This gives an overall total of (3 training + 13 testing records) X 50 virtual surveys X 18 sonification algorithms = 14,400 distinct sonic record classifications.

The result is that the sawtooth waveform sonification using 2 summed waveforms, with the second waveform at 2X frequency, labeled *SawX2* in the bottom of Figure 5, gives the greatest mean accuracy for RandomTree –K=4 –M=1.0 –S=0 of 76.05%, about 9 points above the "All 3 sets" entries of Table 1. This result establishes the viability of using machine listening as an adjunct to human surveys in evaluating sonification algorithms.

Weka's RandomTree-generated decision tree for SawX2 has the following structure, similar in simplicity to the WaveX2 decision tree.

If Centroid < 1764 Hz (.08 * 22,050)

Classify as Set 1. (NOT Set 0 as for WaveX2.)

Else if fft6 centered at 2412 Hz < .02 (NOT .03)

Classify as Set 0. (NOT Set 1 as for WaveX2.)

 # .02 is 3.3 on Fig. 5 log scale.

Else

Classify as Set 2.

Sawtooth sonification shifts the lowest centroid to the mean of Set 1 records, improving the ability to distinguish between Sets 1 and 2. Set 0's fft6 measure for SawX2 has a greater distance from Set 2's fft6 than Set 1-to-Set 2 has for WaveX2's fft6. These simple spectral changes of SawX2 improve mean classification accuracy by 9% over WaveX2 for RandomTree.

5. CONCLUSIONS AND FUTURE WORK

A parallel coordinates plot provides a viable basis for mapping from a data visualization to a data sonification of tuples (records) for the purpose of classifying those tuples. Timbral waveform sonification treats each vertical attribute axis in a parallel coordinates plot as a point in time, and a tuple's meandering path across its attributes' values on these vertical axes as a time-domain waveform. An earlier study has established the superiority of converting a parallel coordinates representation to such a waveform sonification, as compared with mapping attribute values to discrete simultaneous-harmonic intervals or serial-melodic intervals [9]. The waveform technique maps each attribute's value to an inflection point on a waveform.

The present study uncovers advantages in mixing multiple, frequency-shifted copies of a waveform, with high-pass filtering, for human classification of data tuple sounds in terms of their aural proximity to mean reference sounds. The Wave2X algorithm that doubles the frequency spectra of a shifted copy of the generated waveform provides better sonification, as determined by the responses of human sonic survey takers, than alternatives that shift a waveform copy by 4/3 or 1.95 times the original waveform frequency spectra.

Machine listening achieved by using machine learning algorithms to classify sounds in audio data files provides a viable means to evaluate waveform sonification algorithms. *RandomTree* operates by performing random searches of the attribute space in building decision trees. By executing many RandomTree test runs with varying configuration parameters, we have found that the RandomTree decision tree classifier with options –K (number of randomly chosen attributes) = 4, -M (minimum weight of a leaf in the tree) = 1.0, and –S (random seed) = 0, is a classifier that gives the same mean percentage of correct responses as the human survey takers, where the correct classifier responses range between 64% and 65%. Our RandomTree decision trees focus strictly on a waveform's centroid and the seventh bin (fft6 from fft0) in a 64-bin fft from 0 to 22050 Hz.

Two classifiers that are more powerful than RandomTree in this application are K* and AdaBoostM1 with HoeffdingTree. While space limitations preclude detailed discussion of these and other machine learning algorithms applied to the problem, there are some useful points to note. K* is an instance-based learner that does not build a classification tree or other abstract model. It measures distances from data waveform attributes such as centroid, power, rolloff frequencies, and frequency-domain spectral values, to classification reference waveform attributes. By comparing test sounds to training sounds, one at a time, it appears that K* is similar in a general sense to the approach taken by human survey takers.

AdaBoostM1 is an ensemble learning algorithm that builds on the strength of using HoeffdingTree as a base classifier. HoeffdingTree is notable for using small training sets effectively. It is likely that, in addition to RandomTree, K* and AdaBoostM1 / HoeffdingTree will be useful in evaluating timbral sonification algorithms to be investigated.

SawX2 that sums a tuple-derived sawtooth waveform and its frequency-spectra-doubled copy has been the most accurate sonification algorithm so far as analyzed by RandomTree.

Future work includes experimenting with additional waveform forming and filtering options of sonification algorithms, for evaluation using machine learning algorithms, with an eventual goal of re-employing human survey takers to validate the final results of machine listening. Replacing the sweet & sour step function of Figure 3 with a non-linear sonification function for each inflection point in a waveform, such as squaring or using another polynomial function of an attribute value's distance from the Reference Set 0 mean, is an approach that promises to add distinctive sonic marker frequencies for different reference sets. RandomTree's explicit decision trees provide means to identify audio attributes for emphasis. We also plan to sonify additional datasets to establish the general applicability of this approach.

6. REFERENCES

[1] T. Hermann, A. Hunt, and J. Neuhoff (editors), *The Sonification Handbook*, Logos Verlag, Berlin, Germany, 2011.

[2] J. Han, M. Kamber, and J. Pei, *Data Mining – Concepts and Techniques*, Third Edition, Morgan Kaufmann, 2012.

[3] D. Parson and A. Seidel, "Mining Student Time Management Patterns in Programming Projects," *Proceedings of FECS'14*: *2014 Intl. Conf. on Frontiers in CS & CE Education*, Las Vegas, July 21-24, 2012. http://faculty.kutztown.edu/parson/pubs/FEC2189ParsonSeidel2014.pdf.

[4] D. Parson, L. Bogumil, and A. Seidel, "Data Mining Temporal Work Patterns of Programming Student Populations," Proceedings of the 30th Annual Spring Conference of the Pennsylvania Computer and Information Science Educators (PACISE) Edinboro University of PA, Edinboro, PA, April 10-11, 2015. http://faculty.kutztown.edu/parson/pubs/StudentTimePacise2015Kutztown.pdf

[5] A. Inselberg, *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*, 2009 Edition, Springer, 2009.

[6] G. Loy, *Musimathics*, Volume 2, The MIT Press, 2011.

[7] J. Neuhoff, "Perception, Cognition and Action in Auditory Displays," The Sonification Handbook, T. Hermann, A. Hunt, and J. Neuhoff (editors), Logos Verlag, Berlin, Germany, 2011, p. 63-85.

[8] M. Schedel and K. Yager, "Hearing Nano-Structures: A Case Study in Timbral Sonification," *Proceedings of the 18th International Conference on Auditory Display* (ICAD2012), Atlanta, Georgia, June 18 – 22, 2012.

[9] D. Parson, D. E. Hoch, and H. Langley, "Timbral Data Sonification from Parallel Attribute Graphs," *Proceedings of the 31st Annual Spring Conference of the Pennsylvania Computer and Information Science Educators* (PACISE) Kutztown University of PA, Kutztown, PA, April 1-2, 2016.

[10] Machine Learning Group at the University of Waikato, "Weka 3: Data Mining Software in Java", http://www.cs.waikato.ac.nz/ml/weka/, January 2016.

[11] Witten, Frank, and Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, Third Edition, Morgan Kaufmann, 2011.

[12] Kapur, Cook, Salazar and Wang, *Programming for Musicians and Digital Artists: Creating music with ChucK*, Manning Publications, 2015.

[13] W. Malke, *Machine Listening with Very Small Training Datasets*, master's thesis, Kutztown University of PA, January 17, 2017. Please contact the authors for a copy.

[14] R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[15] G. Loy, *Musimathics*, Volume 1, The MIT Press, 2011.

[16] G. Hulten, L. Spencer, and P. Domingos: "Mining time-changing data streams", In: *ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 97-106, 2001.