EIGENRHYTHMS: DRUM PATTERN BASIS SETS FOR CLASSIFICATION AND GENERATION

Daniel P.W. Ellis and John Arroyo
LabROSA, Dept. of Elec. Eng., Columbia University, NY NY USA
`dpwe@ee.columbia.edu, ja2124@columbia.edu`

ABSTRACT

We took a collection of 100 drum beats from popular music tracks and estimated the measure length and downbeat position of each one. Using these values, we normalized each pattern to form an ensemble of aligned drum patterns. Principal Component Analysis on this data set results in a set of basis 'patterns' that can be combined to give approximations and interpolations of all the examples. We use this low-dimension representation of the drum patterns as a space for classification and visualization, and discuss its application to generating continua of rhythms. Our classification results were very modest – about 20% correct on a 10-way genre classification task – but we show that the projection into principal component space reveals aspects of the rhythm that are largely orthogonal to genre but are still perceptually relevant.
Keywords: rhythm, genre, classification, principal components

1. INTRODUCTION

Popular music usually includes a drum 'track' providing the rhythmic backbone of the piece, and the percussion instruments generally play a short pattern that repeats every few beats. This core pattern, along with the rate at which it is played (typically measured in beats per minute, or BPM) constitute a key element in the character of the music.

We are interested in describing and extracting such essential subjective characteristics from music as part of our wider project into music similarity and recommendation [4, 1, 2]. Our previous work has focused on average spectral content, pitches, and chords [15, 13], but has not included explicit rhythm-related features. This paper describes an initial study into extracting and describing this kind of information.

Many previous music information retrieval systems have tapped the rhythm dimension. Tzanetakis et al. employ a small set of rhythm descriptors including BPM and "rhythm strength" [16], and Gouyon et al. show the value of cosine transform coefficients of a time-warped log-inter-onset-interval histogram [6] for classifying dance music genres. There is, however, a gulf between the very large range of possible drum patterns – spanning variations in basic note patterns, accent, and small time shifts ("swing") – and the small number of dimensions desirable in classification and browsing systems.

2. EIGENRHYTHMS

We propose to bridge this gulf using the standard dimensionality reduction tool of Principal Component Analysis (PCA) [3]. An ensemble of data that can be represented as points in a high-dimensional space can be approximated as the weighted sums of a few basis vectors in that space; the covariance matrix of the ensemble provides information about which dimensions are correlated (i.e. exhibit co-ordinated changes), and by finding the eigenvectors of the covariance matrix with the largest eigenvalues PCA finds the basis functions that minimize the distortion of a lower-dimensional representation. Each point in the original high-dimensional space is represented by a smaller number of coefficients, which are the weights applied to each of the principal component vectors to approximate that point. Individual principal components, ordered according to their contribution to the overall distortion, can be interpreted as the main dimensions of variation among the examples in the set.

In this work, we represent drum patterns as a simple two-dimensional surface. The horizontal dimension is time, densely sampled to provide a fine resolution of drum-note events (for the results below, we used 5 ms sampling). The vertical dimension corresponds to the different instruments: we caricature popular music drum tracks as consisting of three instruments, bass drum, snare, and hi-hat, and have one row for each. The values in this surface are pseudo energy envelopes: each beat event is represented by a brief, decaying pulse in the surface. We use half-Gaussians with a standard deviation of 20 ms, which, unlike single impulses, can gloss over small amounts of jitter in the timing of individual beats, while retaining a sharp, well-defined onset. The principal components of these surfaces, the two-dimensional surfaces that can be combined to approximate the entire set, constitute our "eigenrhythms".

Our goal is to produce systems that can be applied to actual recordings, but to simplify the investigation of underlying rhythmic information we sidestepped the stage of extracting drum events from audio by working directly from MIDI files in which note event times and drum voice identities are provided explicitly. We do not consider this a serious limitation given the success reported in automatic drum beat extraction from audio: Real-time extraction of bass drum and snare events was reported by Goto ten years ago [5], and more recent work has included adaptive learning of the different drum sounds [17] among many other refinements.

Our principal motivation is to investigate the viability of low-dimensional descriptions of rhythm patterns, but in order to motivate and evaluate our results we apply the representations to a genre classification task. However, we do not pay much attention to making a careful and comprehensive description of the rhythmic pattern in each individual piece: Since we are interested in the gross behavior of a large collection of different drum patterns, the main thing we want is a large number of diverse patterns extracted from different peices. If, for a particular piece, we only extract one of several patterns used, or even if our extraction algorithm fails and returns a nonsense pattern, it is not of great concern as long as the bulk of the database – the material that the most significant principal components describe – is valid. This is another reason for starting with MIDI: there are tens of thousands of MIDI files readily available on the internet, and we would like to be able to model rhythmic information from collections of this scale without the computational load of processing an equivalent amount of audio. By limiting ourselves to pulling the pattern from a single, small excerpt taken from the middle of each piece, we also avoid the issue of tempo tracking – we assume the tempo is constant over the 10 s excerpts we use, and estimate a single value.

The next section describes in more detail our method for finding the "eigenrhythm" drum pattern principal components. Section 4 presents the results of our preliminary application to 100 pieces, giving both the eigenrhythms and describing the classification experiments. We discuss some other possible applications, including resynthesizing patterns from arbitrary points in rhythm space, and present our conclusions in section 5

3. METHOD

To apply PCA, we must generate a collection (ensemble) of drum patterns where corresponding beats are aligned in each item. To do this, we must estimate the pattern length in each original drum track (i.e. its BPM), and the position of one reference time point i.e. one pattern-initial downbeat within the excerpt. Given these values, we can extract a fixed number of beats, starting at a downbeat, from each track, stretch or compress them to a single nominal BPM of 120, to form a single entry in our data matrix. The first step, however, is to convert the raw MIDI data into our basic drum pattern time-channel surfaces.

3.1. Preprocessing of MIDI data

We use publicly-available tools to read General MIDI files (GM) culled from the internet into Matlab [8]. In the GM standard, channel 10 is devoted to drum sounds, with each MIDI note, normally used to specify the different pitches, corresponding to a different pre-defined drum sound. We built a map to convert the 85 common voices to our three classes: bass drum, snare, or hi-hat. The vast majority of popular music drum patterns consist only of these three voices. Tom-toms, crash cymbals, and other exotic sounds were discarded (mapped to null). The MIDI velocity parameters, which can be used to convey amplitude accents, were ignored in this work, as were the note offset times. Instead, each onset time resulted in a short, decaying envelope element being added in to appropriate voice's overall time envelope, sampled at 200 Hz. In lieu of a more sophisticated approach, we initially extracted 10 s of drum track starting 30 s into each GM file. An example of such a pattern (from a MIDI replica of "The Next Episode" by Dr. Dre) is shown in the first pane of figure 1.

3.2. Pattern period estimation

Scheirer [12] contrasts zero-phase autocorrelation tempo period estimators with his bank of resonators which indicate both the dominant period and the timing of energy peaks within each channel. However, because we wish to use a more complex approach to downbeat detection, we can use simple autocorrelation to first obtain several period estimates, leaving the downbeat identification (and choice among the period estimate) to a subsequent stage. The second pane of figure 1 shows the positive-lag half of the autocorrelation of the extracted drum-pattern surface shown in the first pane: each of the three tracks (bass drum, snare and hi-hat) first has its total energy equalized (to reduce the influence of the most active voice, usually the hi-hat), then their individual autocorrelations are simply summed. In this example, we see the strongest peak at a lag of around 1.1 s, corresponding to 98 BPM. The next highest peaks are the higher-order multiples at 2, 3 and 4 times this basic period.

In this case, the 98 BPM peak corresponds to the subjective period for this pattern, but in general, the highest peak is not always the best period, and there may be strong peaks at subdivisions as well as integer multiples of the key period. We choose among these by considering each of the N highest peaks from the autocorrelation (where $N = 4$ in the results presented here), and keeping the period that gives the highest normalized cross-correlation in the downbeat estimation, described next.

3.3. Downbeat Location

To get sensible results from PCA, the different patterns in our ensemble must not only have the same tempo, but must be somehow 'lined up' to have equivalent beats at the same time. Although this concept is not well-defined, in many cases it is possible to identify a particular point in a

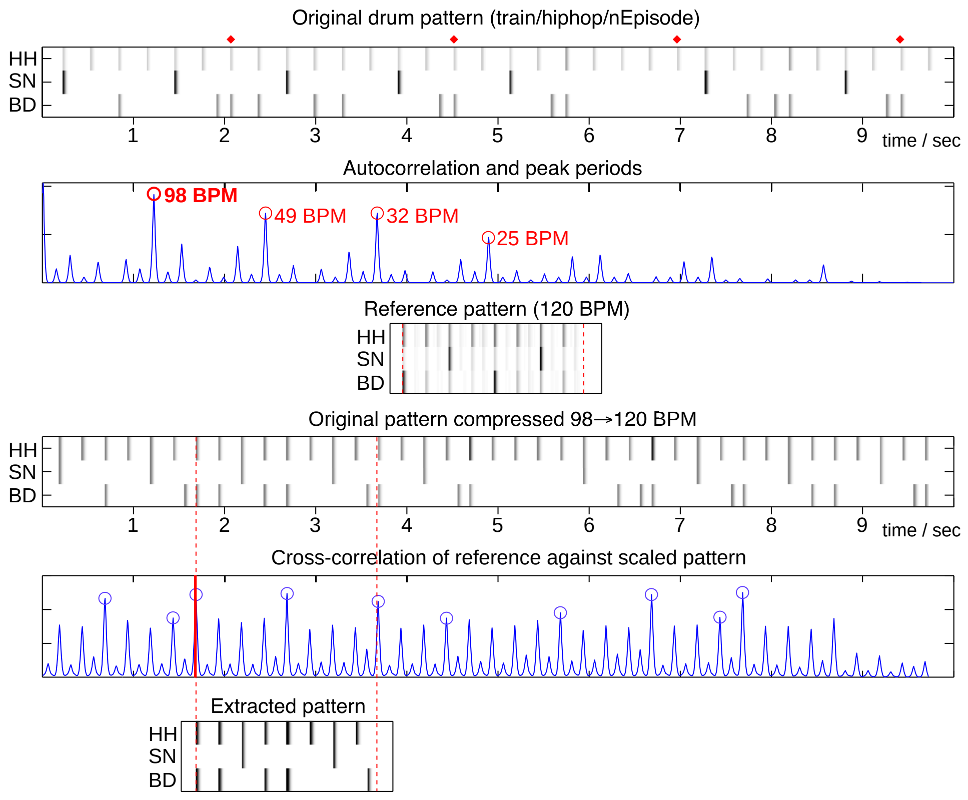

Figure 1. Tempo estimation and downbeat detection. The top pane shows the original pattern extracted from an example MIDI file after mapping into bass drum, snare, and hi-hat; red dots above the pattern indicate hand-marked downbeats. The second pane shows the autocorrelation of this pattern, with the four highest peaks circled and labeled with their equivalent BPMs. Below that is the reference pattern (a grand average of aligned patterns), which is cross-correlated against the original pattern rescaled to 120 BPM (in this case, assuming the 98 BPM peak is valid) to give the fifth pane. The largest peak in this cross-correlation gives the downbeat hypothesis, and leads to the extracted pattern in the bottom panel which then becomes part of the aligned pattern ensemble.

looping drum pattern as the ‘beginning’, and our goal is to locate this point. Regardless of its interpretation, we need some way to choose a unique anchor point in each pattern: if our ensemble includes an arbitrary circular time shift to each pattern, the principal components will be meaningless.

Our approach is to define a reference pattern, consisting of some simplified version of what we are hoping to find, and to cross-correlate this template against the input patterns once their tempo has been normalized. If the input pattern contains that exact subsequence, the cross-correlation will peak at the time-skew that aligns them. Even if the ideal pattern does not occur exactly in the input pattern, the highest peak in the cross-correlation shows the time offset within the longer segment that begins the segment with greatest similarity to the reference pattern, which is an unambiguous anchor point, and gives us an appropriate alignment of ‘maximum similarity’ for extracting a segment to use for the PCA.

For each of the N period hypotheses extracted from autocorrelation, we first time-scale the original MIDI data so that, if the hypothesis is correct, the new note sequence will be at 120 BPM, the tempo of the reference pattern. After finding the cross-correlation peak for the surface derived from that time-compressed or -stretched version, we make a note of the peak cross-correlation value as well as the time offset where it occurs. We normalize the cross-correlation by the energy of the input pattern within a sliding window of the same length as the reference pattern, so the cross-correlation values are always correctly normalized and can reach unity only when reference and input exactly match. We calculate the cross-correlation only for points where there is full overlap between the short reference pattern and the longer scaled input pattern.

We then choose among the BPM hypotheses the one that gave the highest peak cross-correlation value i.e., from among the period hypotheses suggested by the autocorrelation, the temporal scaling of the original input pattern that results in a pattern most similar to the reference pattern appearing. Over-estimates of the original pattern’s period (i.e. picking the 49 BPM peak in the example) will compress more points into the fixed-length segment in the temporally-scaled pattern; while this may lead to more overlap with the peaks in the reference pattern, the extra input notes will lead to a high average energy, so the normalized cross-correlation value will be hurt. Period estimates that are too short (high BPMs) will have normalized versions that are too stretched out in time and are unlikely to have enough points in common with the reference to achieve a high cross-correlation. Thus, the cross-correlation finds the downbeats and chooses the best-matching tempo estimate in a single stage.

The reference template we use is actually the average of all the normalized patterns emerging from our analysis, but there is a circularity because we need to perform the downbeat alignment before we can calculate this average. To bootstrap, we took a very simple prototype pattern, alternating bass drum and snare with an eighth-note hi-hat pulse, then successively aligned our patterns, formed their average, and re-calculated the downbeat positions using this new average as reference. Once the downbeat positions match in two successive iterations, the system has converged and there will be no further changes in later iterations. We observed convergence within 5 cycles.

The grand average reference pattern template is shown in the third pane of figure 1, along with one of the time-scaled drum patterns, in this case for the correct 98 BPM hypothesis; the fifth pane shows the results of cross correlation, with the top 10 peak values circled; for now, we consider only the top value in the cross-correlation and use that as our downbeat, assuming that it gives the largest peak value across all the BPMs being considered.

Finally, we extract a short segment from the 120 BPM-scaled input patterns, corresponding to the 4-beat segment of the reference template, and pass this forward to the principal component analysis. We take four beats because 2 beats (e.g. a single bass drum/snare alternation) seemed too short to capture much interesting structure in the pattern; after reviewing the training examples, many of which contain 8- or 16-beat basic patterns, there could be good reason to use a longer excerpt, although this might necessitate a lower temporal resolution to our surfaces in order to keep our PCA computationally tractable.

3.4. Principal Component Analysis

The processing so far gives, for each input drum track, one 2 s excerpt of the rhythm pattern after normalization to 120 BPM (i.e. four beats in total), starting at a downbeat defined by the best alignment to a reference rhythm. With three voices and a sampling rate of 200 samples per second, this is a 1200 point feature for each piece. We simply stack these vectors for each of our examples, calculate and subtract the mean pattern (which is just the reference pattern used in extraction, once the analysis has converged) and apply singular value decomposition to the covariance matrix of this data to find the eigenvectors. In our experiments, we used just 100 MIDI tracks, giving a maximum of 99 nonzero eigen dimensions, although our goal is in using many fewer dimensions than this to get at the 'essence' of the rhythms. The projection of each rhythm pattern into a subset of the most significant principal components provides for classification (e.g. by nearest neighbor), and the space provides interesting interpolations; by compressing the dimensionality to maximally preserve the structure in the real rhythms, we have a space where unnatural rhythms most likely cannot be represented, and all points correspond to reasonable-sounding rhythms.

4. RESULTS

For our tests, we collected a set of 100 MIDI tracks, arranged as 10 examples for each of 10 genres. Our intention was to provide a reasonable selection of the kind of music typically heard on contemporary popular music radio. The genres were also defined with an eye to distinctions that could possibly be discerned in the rhythm patterns (according to our intuitions) and also the availability of enough suitable MIDI files. We verified that each of the files we selected was a well-produced replica and a satisfactory representative of its class, but did not use any more specific criteria in selecting them. The ten categories can be seen on the axes of the confusion matrix in figure 2.

To evaluate the raw tempo and downbeat extraction, we auditioned each tracking result by resynthesizing the original drum pattern along with added tone pips indicating the system's chosen downbeats and cycle length. In two of the cases the arbitrary initial note extraction returned irregular drum patterns for which no period could be decided. In nine of the remaining 98 cases (9.2%), the period chosen by the system was wrong, almost always half the length (i.e. tempo twice as fast) as the perceived period. Where the tempo was correct, about half the tracks had patterns of 4 or 8 beats (rather than the basic 2-beat bass/snare pattern), and of those, approximately half (25 out of 53) had the downbeat in the right point within that sequence. In the others, the downbeat was shifted by an even number of beats. This is a secondary error, since the extracted pattern was basically appropriate, but it would make for a better interpolation space if the automatic downbeat placement could come closer to subjective impression. We return to this in the discussion.

4.1. Classification task

The 100 extracted patterns, each represented by a 1200-point envelope, were then fed to PCA to extract the eigenrhythms; the mean pattern and top five eigenrhythm bases are illustrated in figure 3; 25 dimensions are required to explain 90% of the variance. Restricting ourselves to the top N eigenrhythms gives an N-dimensional projection of the set of rhythms that minimizes squared-error distortion, and which can also be seen as a kind of generalization, smoothing away 'insignificant' differences between patterns. This reduced space can be used for classification, for instance by treating each of the patterns in turn as unknown and classifying it on the basis of its k nearest neighbors (k-NN classification). Although this is not as good a predictor of classifier success as using test data

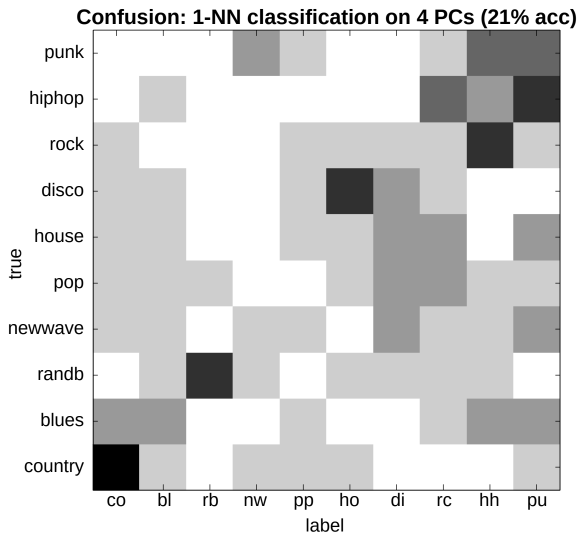

Figure 2. Confusion matrix for genre classification based on eigenrhythms. Classification was based on the single nearest neighbors according to the four top eigenrhythms. Overall classification accuracy was 21% in this case, compared to 10% from random guessing.

separate from the data used in deriving the eigenrhythm space, we note that the genre labels were not involved in that stage i.e. the PCA 'model' does not encode prior knowledge of the true class of the test examples.

We performed this classification and searched over the number of PCA dimensions (from 2 to 40) and the number of neighbors to use in the k-NN classification. Our results were generally weak; one of the best performing combinations was the simple case of using 4 dimensions and classifying according to the single nearest neighbor. The 10×10 confusion matrix for this case is shown in figure 2. Although the overall classification accuracy is 21%, this is significantly better than random guessing (which would give 10%), and the confusion matrix reveals some cliques of greater discrimination, including $\{$country, blues$\}$ and $\{$rock, hiphop, punk$\}$. Rhythm and Blues (randb) is recognized correctly 4 out of 10 times, which is also how often disco is recognized as house. All of these details seem to make sense in view of the musical character of the different classes.

4.2. Eigenrhythms

For a greater insight into the classification performance, and to see what the "eigenrhythm" concept has actually captured, it is interesting to look at the top eigenrhythms individually, as in figure 3. The top-left panel shows the overall mean pattern, which is subtracted from every pattern prior to the eigen analysis; this pane is nonnegative, with white showing regions of no energy and darker shades of gray indicating progressively more intense beats. The remaining patterns have, in general, both positive and negative portions, and can be associated with positive or negative weights to add to or subtract from different beats in the mean pattern – i.e. increasing or reducing the contrast between their positive and negative extrema. These patterns are shown with positive excursions in green, negative values in red, fading to white as the value tends to zero (with apologies to those viewing this paper in monochrome!). Eigenrhythm 1 is mainly positive, following the mean pattern, but includes emphasis of the 16th notes (at samples 25, 75, 125 etc.) in the snare and hi-hat, the snare beats on the eighth notes at samples 50, 150 and 350, and the bass drum simultaneous with the main snare beats at samples 100 and 300 (quarter note beats 2 and 4).

Eigenrhythm 2 provides contrast between the eighth-note hi-hats, and the snare and bass hits on beats 2 and 4 along with a snare eighth-note 'echo' at samples 150 and 350. The third basis contrasts hi-hat beats on the 16th note 'off' beats with a simple quater-note rhythm, so a negative coefficient here will introduce a double-speed hi-hat, and a positive weight gives half-speed. Basis 4 contrasts off-beat bass drums at samples 50, 150, 250 and 350 with hi-hat beats alongside the main snares at beats 2 and 4, as well as including some fast snare beats between samples 200 and 300 and some evidence of 6/8 patterns with hi-hat features around sample 67, 167, 267 etc. (i.e. two-thirds of the way through each quarter-note beat). The final eigenrhythm in the figure contrasts snare and bass on beats 2 and 4, and also can be seen to provide for more complex structures in hi-hat and bass drum.

Overall, we begin to see that the individual eigenrhythms are encoding particular features of the different rhythmic patterns, more about the character of the piece than its genre, but at the level of groups of notes rather than individual events. Although genre is only weakly predicted by nearest neighbors in the eigenspace, we can ask if there are other perceived properties being preserved. Figure 4 is relevant here, which shows all the training points projected onto the first two eigenrhythms. Although the names are hard to read when the text overlaps, what stands out are the clusters, including those around (2, 1) and (4, -1). (The polarity of eigenrhythm 1 has been flipped in this image for clarity). Listening to the individual tracks involved, the first cluster consists of straight-ahead 4/4 patterns with eighth note hi-hat patterns, snare on beats 2 and 4, and some variations in the bass drum within the basic eighth-note grid. Patterns in the second cluster, by contrast, have the same basic bass drum on beats 1 and 3 with snare on beats 2 and 4, but have a 6/8 'syncopated' rhythm in the hi-hat, or a simple quarter-note hi-hat beat. Thus, once the tempo variation is removed, we find that the eigenrhythm space does indeed cluster drum patterns with clearly discernible perceptual similarities.

5. DISCUSSION AND CONCLUSIONS

Our approach can be contrasted with other work that looks at the detail of rhythm patterns. Laroche has presented a system able to extract 'swing' represented by slight systematic timing shifts of beats 2 and 4; his approach en-

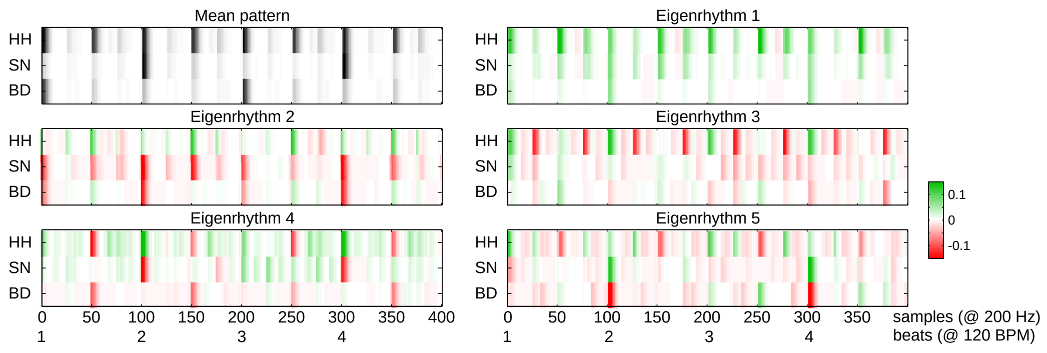

Figure 3. Mean drum pattern and top 5 eigenrhythms.

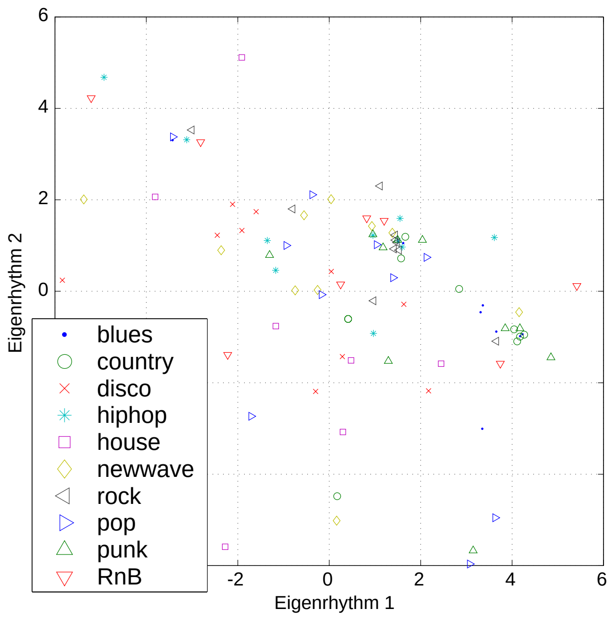

Figure 4. Projection of all tracks onto first two eigenrhythm principal components. Each point's shape and color indicate the ground-truth genre according to the legend.

codes more musical knowledge (with a correspondingly narrower applicability) than we wished to use [9]. Paulus and Klapuri present a system for comparing the rhythm patterns between two different pieces, overcoming variations in drum sounds with cepstral normalization and minor timing differences through dynamic time warping, but they do not attempt to build a parametric space of rhythmic variants [11].

One natural application for the eigenrhythm representation is for the generation of rhythm patterns that interpolate between the different points, e.g. in the plane of figure 4. We have built a crude Matlab interface to synthesize the rhythm patterns corresponding to any value of the first eight eigenrhythms allowing us to investigate the space, but we hope to produce something a bit more interactive.

In evaluating our current system, it became clear that the 2 s excerpts used in modeling were too short. Many tracks had drum patterns that repeated at a larger scale that this, and the system had little chance of finding the appropriate downbeat within such patterns when only part of the cycle is modeled. Even so, the downbeat detection appears to require a more sophisticated approach. Lacking some absolute principle to decide where the cycle starts, we believe that our technique of matching a model derived from actual data is the right basic approach, but it may need to be seeded with ground-truth on actual downbeats for at least some of the training examples, and could require a family of prototype patterns (e.g. finding the temporal alignment that supports the best fit from a set of eigenrhythms) rather than relying on a single, average template.

As noted above, our initial interest was simply to collect a large body of drum patterns to see what the principal components would be like. However, a more careful musical information extraction technique would consider the entire drum track of a piece, looking for the regularly-repeating patterns and perhaps also modeling the less repetitive breaks and ornamentations. We think the eigen analysis should also be applicable to drum breaks, if they can be effectively extracted, although because their duration is less constrained some kind of sequential structure (such as a hidden Markov model) might be appropriate. One could imagine a Markov model where each state is represented by values or a distribution in eigenrhythm space, and transition probabilities encode the likely evolution of the entire piece.

There are many details even in the work we have described that deserve closer examination. Where we have investigated alternatives at all our main metric has been the genre classification accuracy, which is so low as to be suspect and doesn't directly address our main interest of defining a space of ‘good’ drum patterns. One idea is to replace the asymmetric envelopes used to represent each drum event with a smoother shape like a full Gaussian. This might allow small time shifts (like Laroche's ‘swing’) to be effectively encoded by eigenvectors that can perform a linear cross-fade between two nearby peaks.

If, on the other hand, we wished to pursue the genre classification application, we could look at more discriminative ways to define our basis functions, such as using Linear Discriminant Analysis (LDA) in place of PCA. LDA is another procedure for finding a low-dimensional projection of a dataset, but it uses class labels associated with each training pattern to find projections that maximally separate classes, to be the most useful in classification [3].

There are several other interesting projection algorithms to consider. Independent Component Analysis (ICA) finds basis projections that are not orthogonal but which maximize the statistical independence of the projected coefficients, which can be a more semantically relevant decomposition [7]. Non-negative Matrix Factorization (NMF) finds linear basis sets where all the coefficients are positive or zero, so each pattern is approximated by a process of 'adding in' parts, rather than the balancing contrasts seen in our eigenrhythms [10]. When the underlying dataset is intrinsically nonnegative, as in our case, this can be an interesting alternative transformation; some previous applications to musical audio are reported in [14].

In conclusion, we have introduced a new representation for the complex but constrained class of popular music drum patterns, and derived our basic eigenrhythm patterns by scaling and aligning a corpus of drum tracks from real pieces, encoded as MIDI files. We hope to use larger datasets and a deeper analysis to come up with a more general model of the stylistic variations in rhythm patterns, and we hope to be able to train from, and apply to, actual recorded waveforms.

6. ACKNOWLEDGMENT

This work was supported in part by the NSF under grant no. IIS-0238301, and by a grant from Google. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

7. REFERENCES

[1] A. Berenzweig, D. P. W. Ellis, and S. Lawrence. Anchor space for classification and similarity measurement of music. In *ICME 2003*, 2003.

[2] A. Berenzweig, B. Logan, D. P. Ellis, and B. Whitman. A large-scale evalutation of acoustic and subjective music similarity measures. In *Proc. Int. Conf. on Music Info. Retrieval ISMIR-03*, 2003.

[3] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, New York, 2001. 2nd ed.

[4] D. Ellis, B. Whitman, A. Berenzweig, and S. Lawrence. The quest for ground truth in musical artist similarity. In *Proc. Int. Symposium on Music Inform. Retriev. (ISMIR)*, pages 170–177, 2002.

[5] M. Goto and Y. Muraoka. A beat tracking system for acoustic signals of music. In *ACM Multimedia*, pages 365–372, 1994.

[6] F. Gouyon, S. Dixon, E. Pampalk, and G. Widmer. Evaluating rhythmic descriptors for musical genre classification. In *Proc. 25th Intl. AES Conf.*, London, 2004.

[7] A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4–5):411–430, 2000.

[8] F. Kurth. Matlab-MIDI tools, 2004. `http://www-mmdb.iai.uni-bonn.de/download/matlabMIDItools/`.

[9] J. Laroche. Estimating tempo, swing and beat locations in audio recordings. In *Proceedings of the 2001 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Mohonk NY, 2001.

[10] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[11] J. Paulus and A. Klapuri. Measuring the similarity of rhythmic patterns. In *Proc. ISMIR-02*, Paris, 2002.

[12] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *J. Acoust. Soc. Am.*, 103:1:588–601, 1998.

[13] A. Sheh and D. P. Ellis. Chord segmentation and recognition using em-trained hidden markov models. In *Proc. Int. Conf. on Music Info. Retrieval ISMIR-03*, 2003.

[14] P. Smaragdis and J. C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Proc. IEEE Workshop on Apps. of Sig. Proc. to Acous. and Audio*, Mohonk NY, 2003.

[15] R. J. Turetsky and D. P. Ellis. Ground-truth transcriptions of real music from force-aligned midi syntheses. In *Proc. Int. Conf. on Music Info. Retrieval ISMIR-03*, 2003.

[16] G. Tzanetakis, G. Essl, and P. Cook. Automatic musical genre classification of audio signals. In *Proc. ISMIR-01*, Bloomington IN, 2001.

[17] A. Zils, F. Pachet, O. Delerue, and F. Gouyon. Automatic extraction of drum tracks from polyphonic music signals. In *Proc. WEDELMUSIC-02*, December 2002.