BEAT TRACKING USING AN AUTOCORRELATION PHASE MATRIX

Douglas Eck

University of Montreal
Department of Computer Science
Montreal, Quebec H3C 3J7 CANADA
douglas.eck@umontreal.ca

ABSTRACT

We introduce a novel method for estimating beat from digital audio. We compute autocorrelation such that the distribution of energy in phase space is preserved in a so-called Autocorrelation Phase Matrix (APM). We estimate beat by computing individual APMs over short overlapping segments of an onset trace derived from audio. Then an adaptation of Viterbi decoding is used to search the APMs for metrical combinations of points that change smoothly over time in the lag/phase plane. Because small temporal perturbations are seen as local movements on the APM, the Viterbi search can be bounded using a small 2D Gaussian window. The resulting algorithm jointly estimates tempo, meter and beat. As is always the case with Viterbi decoding, an online version is possible although best performance is achieved offline. We report results on an annotated dataset of 60-second musical segments.

Index Terms— Correlation, Viterbi decoding, beat induction, Autocorrelation Phase Matrix

1. INTRODUCTION

One challenge in estimating beat is that there is no single correct answer. That is, several good beat assignments can exist in parallel. Furthermore, beat can shift both in time (e.g. between different levels of the metrical hierarchy) and in phase (e.g. from unsyncopated to syncopated). Finally, beat can change in the middle of a piece, such as when the meter and tempo shifts are encountered. These observations suggest that beat estimation is a process where multiple hypotheses about beat are considered in parallel and where it is possible to switch between these hypotheses.

A full full survey of computational beat models is impossible due to space constraints; we mention two approaches here due to their relevance to our approach. Scheirer [1] uses comb-filter resonator banks to estimate tempo. He argued that phase could be recovered via an examination the internal states of the bank delays. With reliable phase information it should be possible to estimate beat. The model from Klapuri et al. [2] uses a similar two-stage estimation process where tempo prediction is followed by phase recovery. To estimate tempo, the model integrates evidence at tatum, tactus and measure levels. This is achieved using a Hidden Markov Model (HMM) with hand-encoded transition probabilities set using prior knowledge about human tapping rates. Viterbi decoding—a technique common in speech recognition [3]—is employed to find the most likely sequence of HMM states over time.

Our work is perhaps most similar to that of Klapuri et al. in that we integrate evidence at different timescales and we use Viterbi decoding to find an optimal sequence of predictions. What differs in our work is (a) instead of considering only tatum, beat and tactus evidence, we consider arbitrarily-deep metrical hierarchies, (b) we use autocorrelation instead of comb filtering and (b) we estimate tempo, beat and meter in a single step.

2. ALGORITHM

In previous work [4] we described a data structure—the Autocorrelation Phase Matrix (APM)—that allows for the computation of autocorrelation without loss of phase information. To sum, instead of storing the results for a particular lag-k correlation in a single value, the results over time (t) are stored in row k of the APM. The column index for storing the correlation for $x(t)x(t+k)$ is indexed using $t \bmod k$. By summing row-wise across the matrix, standard autocorrelation is recovered. More importantly phase information about the distribution of autocorrelation energy can be used to perform beat tracking.

Background: As in most if not all beat estimation models, the first step is to extract an onset trace using an onset detection function. In developing the algorithm we tried several existing approaches including that of Scheirer [1]. See Bello et al. [5] for an overview. In the end none of the complex approaches worked better than computing a 1024-point Constant-Q spectrogram, differentiating the log magnitude of the spectrogram over time and summing over frequency to yield an envelope. Figure 1 shows the signal, the envelope and the autocorrelation of the envelope. We compute the spectrogram such that a frame comprises 10msec of audio data with 512-points of overlap. The hop size h of the spectrogram is calculated as follows: $h = fs_{orig}/fs_{env} + 512$ where fs_{orig} is the sampling rate of the signal, fs_{env} is the desired sampling rate for our envelope (100Hz) and 512 is the number of overlapped points. In the figure the actual tempo of the song (484ms; 124 BPM) is marked with a vertical line. This tempo and its integer multiples are also marked with stars. Levels in the metrical hierarchy (periodicity of quarter note, half note, etc.) are marked with triangles.

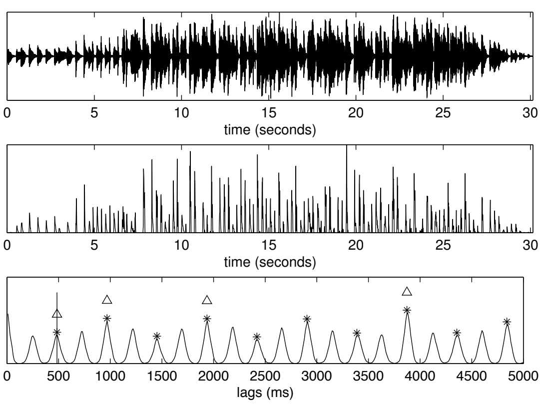

Fig. 1. Timeseries (top), envelope (middle) and autocorrelation (bottom) of a ChaChaCha from the ISMIR 2004 Tempo Induction contest (Albums-Cafe_Paradiso-08.wav). A vertical line marks the actual tempo (484 msec, 124bpm). Stars mark the tempo and its integer multiples. Triangles mark levels in the metrical hierarchy.

Autocorrelation Phase Matrix (APM): The APM is an extension of standard autocorrelation. For each lag k of interest, the APM stores intermediate results of autocorrelation in a vector of length k such that the results of the dot product from the autocorrelation are distributed into that vector by their phase (ϕ). Phase is constrained such that for all k, $\phi < k$ hence resulting in triangular matrices. Observe that the APM (here denoted as P) preserves the distribution of autocorrelation energy in phase space.

$$P(k, \phi) = \sum_{i=0}^{(N/k)-1} x(ki + \phi)x(k(i + 1) + \phi) \quad (1)$$

At the same time, a counter matrix C allows for the computation of unbiased autocorrelation: $C(k, \phi) = N/k$.

For applications such as beat induction, it is useful to have a causal model so that processing can be done online. The pseudo-code in Algorithm 1 describes one simple causal version of the APM. Our own Matlab/C++ implementation is implemented using an optimized version of this algorithm.

Algorithm 1 Update for single timestep t.

Input: X {buffered signal}
Input: K {set of m lags; max lag is n}
Input: P, C {APM and counter; size [m,n]}
for $i \leftarrow 0$ to $m - 1$ **do**
  if $t >= K[i]$ **then**
    $\phi \leftarrow \text{mod}(t, K[i])$
    $P[i, \phi] \leftarrow P[i, \phi] + X[t] * X[t - K[i]]$
    $C[i, \phi] \leftarrow C[i, \phi] + 1$
  end if
end for

The key idea behind the APM is its ability to reveal repeating phase-correlated structure in a signal. This can be seen in the row-wise repetition of structure (Figure 2), which was computed from the same ChaChaCha song used above. Autocorrelation

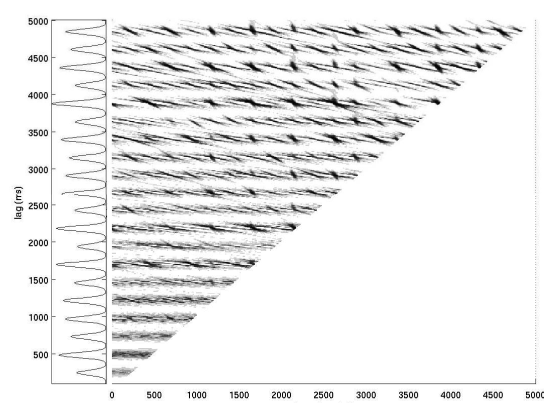

Fig. 2. The APM for Albums-Cafe_Paradiso-08.wav, the same song as shown in Figure 1. On the left the autocorrelation is recovered by summing rows in the matrix.

$a(k)$ and unbiased autocorrelation $a'(k)$ can be recovered from the APM by summing all phase values for each lag, where the unbiased version is normalized using counter matrix C. Let $P' = P/C$ (where "/" is point-wise division) be the unbiased APM. Then:

$$a(k) = \sum_{i=0}^{k-1} P(k,i) \quad (2)$$

$$a'(k) = \frac{1}{k}\sum_{i=0}^{k-1} P'(k,i) \quad (3)$$

Estimating Beat: Beat is estimated by looking for persistent high-magnitude $< phase, lag >$ values in the APM. It is possible to do this using online updating of a single APM. In this case a decay term would enforce "forgetting", thus keeping the APM from saturating over time. However, it is a challenge to compute an optimal trajectory through time using this updating scheme. Instead we compute APMs over overlapping segments of the signal[1]. We can then use efficient Viterbi decoding to compute an optimal trajectory over time through the APMs. Once we have a generated this optimal sequence, beats can be generated using the predicted $< phase, lag >$ values.

Viterbi decoding [3] is a two-pass process over a lattice of states. In our case presume that we have computed a sequence of unbiased APMs $P'_1 ... P'_j$ over short segments. To compute a Viterbi decoding, we must know the probability of transitioning from any $< phase, lag >$ state in P_j to any $< phase, lag >$ state in P_{j+1}. Viterbi decoding can then be computed using a two-pass process. In the forward pass, two values V_j (value) and B_j (backtrace) are computed for each $< phase, lag >$ state in each of the j APMs. Value V_j stores the optimal value for a state given previous context. Backtrace B_j stores which previous state yielded this optimal value. The optimal sequence is computed using a second pass that moves backwards in time starting with the final APM P'_j and following the index stored in B_j to select the optimal value for P'_{j-1}. Viterbi decoding is used in speech recognition for finding an optimal sequence of phonemes for an utterance. One advantage of Viterbi decoding is that it is capable of switching rapidly in response to changing evidence. An online version of Viterbi decoding can be computed with some loss in performance.

[1]Segment length and overlap amount are hyper-parameters whose values did not prove to be overly-important to performance. We use a segment length of 5 seconds with an overlap of 2.5 seconds.

In general Viterbi decoding is slow to compute for an APM, which has $\sim N = 10,000$ states. The slow step is the forward pass where for each state we must consider which *previous* state yielded the best results. This yields a complexity of N^2 for each step j. We can greatly lower this complexity by implementing a local smoothness constraint. Observe that changes in beat resulting from tempo variation are generally small in magnitude. On the APM these small timing perturbations are seen as *local* movement on the APM. That is, slight changes in tempo yield small movements up and down (in lag; relating to tempo shift) or left and right (in phase; relating to jitter). We can take advantage of this local geometry to impose a constraint on the search done in the Viterbi forward step. Specifically we use a small (e.g. $w = 11$) 2-dimensional Gaussian highest at its center and tailing off to some base probability σ_{base} to be used as the probability for all states outside of the Gaussian. We compute the *global* maximum state value for an entire APM P'_j and multiply it by σ_{base}. For each state in the APM we only need to compute the value V_j for the values inside of the Gaussian window. Yet it remains possible to transition away from the Gaussian window by following the global maximum. This yields a tractable and, in practice, well-performing Viterbi decoding procedure that requires $N + Nw^2 \sim= 100K$ operations per step rather than $N^2 \sim= 1M$ operations per step, and runs well in Matlab in a few seconds.

Integrating Metrical Evidence: In practice the algorithm described above does not work very well for some performances, especially those lacking percussive rhythm instrumentation. One problem is that at any single level of the metrical hierarchy (a row in the APM) there can be considerable noise. This noise can be lowered significantly by incorporating evidence from multiple levels in the metrical hierarchy. In our simulations we considered four meters: $2/4, 3/4, 4/4$ and $12/8$ though others are possible. Evidence from the different levels suggested by these meters were incorporated by adding in phase-aligned values from different metrically-aligned levels of the APM. Thus a single APM is transformed into several (four in our case) "maps" that store in a single $< phase, lag >$ state the original state value plus the phase-aligned value for the subdivided lag and the sum of phase-aligned value*s* for the super-division. Viterbi alignment can be performed individually over each of the maps and a single winning meter chosen (as is done for the results reported here) or can be performed over the combined state space of these four maps, allowing for switching between meters in the middle of a performance.

3. RESULTS

We present beat estimation results using the 220-song annotated database from [6]. This database spans six styles including Dance (N=40), Rock/Pop (N=68) Jazz (N=40) Folk (N=22) Classical (N=30) and Choral (N=22) and offers two metrical levels of beat annotation.

In Figure 3 we report an error measure from Dixon [7]: $D_{acc} = \frac{n}{n+F^{-}+F^{+}}$ where n is the number of matched pairs (within $\pm 70ms$), F^{+} are false positives and F^{-} are false negatives. The values reported are taken from the best-performing hierarchical level of the best performing meter for each song. The mean $Dacc$ values are in general good, especially if we ignore the (very difficult) Choral pieces. The global median is significantly higher than the global mean, indicating that the model failed catastrophically on a few songs, lowering the mean but leaving the median high. Note that this is preliminary work. We are currently in the process of implementing other models and other error measures for better comparison.

Genre (N)	D_{acc} mean	D_{acc} median
Dance (40)	0.91	0.98
Rock/Pop (68)	0.80	0.94
Jazz (40)	0.76	0.92
Folk (22)	0.62	0.59
Classical (30)	0.60	0.57
Choral (22)	0.20	0.17
Not Choral (200)	0.76	0.93
All (222)	0.71	0.88

Fig. 3. Hainsworth dataset results. See text for description.

4. FUTURE WORK AND CONCLUSIONS

Though the APM has been shown to be good at predicting tempo [4], the research presented here is the first to be done on beat estimation. Thus there are many directions for future research (some of them already underway): first we can replace the individual Viterbi decodings over specific meters with a single Viterbi decoding that searches all meters at once. This will give us a principled way to select among meters. It will also allow the model to switch among meters during a performance, something that is impossible now. Second, we can replace the onset detection function with a richer (numeric) representation of the signal, allowing us to, e.g., track pitch correlations over time.

We have demonstrated that the Autocorrelation Phase Matrix (APM) can be used for beat estimation. Despite the high-dimensionality of the APM, it is possible using a prior assumption that tempo perturbations are small over time to perform an efficient Viterbi decoding over the state matrix. One advantage of our approach is that it fits in the general framework of correlation-based analysis and can thus be extended to other vectorial representations of audio, including those which represent pitch. The error rates we report are promising but far from conclusive. However we consider the performance of the model to be good enough to warrant further research.

5. REFERENCES

[1] E. Scheirer, "Tempo and beat analysis of acoustic musical signals," *Journal of the Acoustical Society of America*, vol. 103, no. 1, pp. 588–601, 1998.

[2] A. Klapuri, A. Eronen, and J. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Trans. Speech and Audio Processing*, vol. 14, no. 1, pp. 342–355, 2006.

[3] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–285, Feb. 1989.

[4] D. Eck, "Finding long-timescale musical structure with an autocorrelation phase matrix," *Music Perception*, 2006, (In press).

[5] J. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler, "A tutorial on onset detection in music signals," *IEEE Transactions on Speech and Audio Processing*, vol. 13, pp. 1035–1047, 2005.

[6] S. Hainsworth, *Techniques for the Automated Analysis of Musical Audio.*, Ph.D. thesis, University of Cambridge, 2004.

[7] Simon E. Dixon, "Automatic extraction of tempo and beat from expressive performances," *Journal of New Music Research*, vol. 30, no. 1, pp. 39–58, 2001.