Mach Learn (2006) 65:485–515
DOI 10.1007/s10994-006-0586-4

Using duration models to reduce fragmentation in audio segmentation

Samer Abdallah · Mark Sandler · Christophe Rhodes · Michael Casey

Received: 27 July 2006 / Revised: 4 October 2006 / Accepted: 5 October 2006 / Published online: 14 November 2006
Springer Science + Business Media, LLC 2006

Abstract We investigate explicit segment duration models in addressing the problem of fragmentation in musical audio segmentation. The resulting probabilistic models are optimised using Markov Chain Monte Carlo methods; in particular, we introduce a modification to Wolff's algorithm to make it applicable to a segment classification model with an arbitrary duration prior. We apply this to a collection of pop songs, and show experimentally that the generated segmentations suffer much less from fragmentation than those produced by segmentation algorithms based on clustering, and are closer to an expert listener's annotations, as evaluated by two different performance measures.

Keywords Segmentation · Duration prior · MCMC · Gibbs sampling · Wolff algorithm

1 Introduction

In this paper, we will chiefly be discussing segmentation of musical audio signals (such as commercial music recordings), where the segments to be generated correspond to

Editor: Gerhard Widmer

S. Abdallah (✉)· M. Sandler
Queen Mary, University of London, Mile End Road, London E1 4NS
e-mail: samer.abdallah@qmul.ac.uk

M. Sandler
e-mail: mark.sandler@qmul.ac.uk

C. Rhodes · M. Casey
Goldsmiths College, University of London, New Cross, London SE14 6NW
e-mail: c.rhodes@gold.ac.uk

M. Casey
e-mail: m.casey@gold.ac.uk

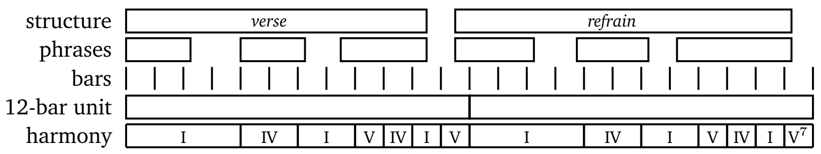

Fig. 1 Example segmentations of a hypothetical fragment of a piece of twelve-bar blues. Each of these represents the structure of different aspects of the piece. See text for further discussion

musical structures at a granularity coarser than individual notes. By 'a segmentation', we mean a system of time intervals, possibly augmented by some auxiliary data for each interval, which correspond to significant physical, sonic, or musical events and structures inferred from the signal. Note that this is more than a collection of boundaries: a segmentation determines a set of boundaries but not the reverse, because, given the possibility of overlapping or nested segments, such a set of boundaries does not indicate which pairs of boundaries should be taken to delimit each segment, and also because a set of boundaries does not contain any extra data that might be associated with each segment.

Musical structures can be found at many levels of detail, some arranged hierarchically and some not. For example, consider Fig. 1, which is a schematic representation of part of a piece in twelve-bar blues form. Metrical structure tends to be strictly hierarchical, with intervals delimited by a regular succession of strong and weak boundaries. Segments at a lower level, e.g. beats, are contained entirely within those at the next level up, in this case, bars. The harmonic structure in this piece is better represented as a succession of (time) intervals labelled by chords. In this example, the chords adhere closely to the basic blues progression, and consequently, the harmonic boundaries align with the metrical boundaries.

Turning to the phrase structure, blues singers and soloists often play ahead of or behind the beat and make use of syncopation, so the phrases might cross metrical and harmonic boundaries, even the boundary of the twelve-bar unit itself, and structures built on phrases (such as perhaps *verse* or *refrain*) will likewise not align with harmonic boundaries.

An application of simple signal processing techniques—for example, measuring spectral differences between frames for onset detection, as in Hainsworth and Macleod (2003)—will yield a segmentation of musical audio; such a 'low-level' segmentation is useful for some purposes, but it does not directly yield the large-scale musical structure of the input.

The segments we aim to produce are those corresponding to structures such as the verse, chorus or introduction of a song; indeed, an integral part of the segmentation algorithms we will be considering is the classification or labelling of the resulting segments such that repeated segments of the same type (say, two instances of the chorus) are tagged with the same label. We shall examine the problems arising from the fact that musical audio has many aspects of structure, and in particular, the confusion between structural levels which is exhibited by segmentation algorithms. If a segmentation algorithm suffers from confusion between these levels, it may exhibit fragmentation or over-segmentation, where the segments generated are too short or otherwise disjoint, corresponding to a level of detail which is finer than the desired one.

1.1 Applications of segmentation

A number of problems brought up by large databases of audio files would be simplified by the existence of a reliable method for automatic segmentation at a meaningful level. Perhaps most obviously, summary generation is strongly connected to segmentation: given a good segmentation at the timescale of a musical section, a good summary (or thumbnail) for an audio track is likely to be given by the most common segment. In Western pop music, this is likely to be the chorus.

A relevant segmentation is closely related to the structure of music; while most music exhibits structure on more than one level (a subject to which we return in Section 8.1), a segmentation even on just one of those levels could be used to inform a user interface for efficient navigation both within songs and across a collection of recordings.

Segmentations can also be used as partial fingerprints for identifying tracks; while the structure of Western popular music is relatively well-established, there is still sufficient scope for variation that the information in a segmentation can help to distinguish between different songs (and likewise to identify different performances of the same song). Finally, segmentation is of assistance in content-based music query systems, both in analysing queries (which might well be entire tracks, in some kind of stylistic proximity search) and to reduce the search space in matching a query against a database.

1.2 The problem of over-segmentation

There are several possible causes of over-segmentation. Logan and Chu (2000) implicitly note, in the context of phrase summarization, one of these causes: a mismatch between the number of distinct segment labels requested and the information in the signal being segmented. If more labels are assigned than are required, then necessarily some of the desired segments will be mislabeled; in addition, each label will model a smaller volume of the feature space, which might cause individual segments to be fragmented to an undesired level of detail. Logan and Chu first generate a segmentation using k-means clustering or HMM (hidden Markov model) state path decoding and then perform an *ad hoc* aggregation to produce the key phrase. Similarly, Peeters, Burthe, and Rodet (2002) note the over-segmentation from a k-means based segmentation, and reduce it by training and finding the state path with an HMM with fewer states than the number of k-means clusters.

Another potential source of over-segmentation is the underlying model of the segments, where the features determining the desired segmentation are themselves not modelled over the right timescale for the desired segmentation. This can occur even when the number of segment labels is appropriate for the level of detail, as disparate parts of different segments might be more similar to each other than adjacent parts in one segment (see Fig. 2 for a schematic illustration). In Abdallah et al. (2005), a

Algorithmic	A	B	A	B	C	B	C	B	A	C	A	C
Desired	A				B				C			

Fig. 2 A schematic illustration of fragmentation in the algorithmic segmentation when the number of segment labels is the same in both algorithmic and desired segmentations

degree of control over this form of excess segmentation was achieved through the use of mean-field clustering (see Section 4.2) and also through the use of large analysis windows, which have the effect of smoothing the landscape and so providing fewer opportunities for oscillations in the dynamics of the model.

However, the solution of using large analysis windows is unsatisfactory, as the precision of any boundary placement will be inversely proportional to the size of the window: the resolution of the segmentation will be limited to the window size. In addition, the smoothing performed by large analysis windows can cause salient short-time details to be missed.[1]

Instead of using large analysis windows, we present in this paper an approach where we define a duration model: a probability distribution for the length of a single segment. This duration model is used to construct a prior probability distribution for segmentations themselves, weighting (with our choice of duration model) the posterior probability for a segmentation away from those with many short segments. This approach does not suffer from the loss of resolution that the approach using large analysis windows does, as it is the duration prior, rather than the length of the analysis window, which tends to encourage averaging of signal features over longer contiguous intervals; the system remains capable of responding to short-timescale signal features.

1.3 Overview

The rest of this paper is organized as follows: we describe previous approaches to musical audio segmentation in Section 2. Section 3 describes the low-level signal processing we perform on an audio signal to generate a time-ordered set of feature vectors. We describe established methods for segmentation by clustering in Section 4, before introducing a duration prior for segment lengths in Section 5 and deriving and explaining our novel algorithm for segmentation given an explicit prior probability in Section 6. We present our experimental evidence in Section 7, before discussing scope for further work and concluding in Section 8. Some notational conventions are given in the Appendix.

2 Previous work on segmentation of music

The approach taken by most of the methods summarized below (as well as our own contribution described here) is to consider some local properties of the signal, analogous to ‘texture’ in vision, and assert that the segments are ‘texturally’ homogenous regions over which distributions of those properties are relatively constant. This implies that the boundaries can only appear where there is a local change in the texture. Although this has been the most common approach to segmentation from audio, it will fail in certain circumstances: consider a song which contains two separated verses in the first half but two consecutive verses in the second. If we successfully identify a local property which corresponds to ‘verseness’, that is, the property is true whenever a verse is in progress, we will detect the first two verses as individual segments but

[1] These details could be accounted for in an ad-hoc way, for example, by quantising or biasing boundary positions to onset times found by a separate higher-resolution onset detector.

the other two will be merged into one long verse segment, even if there are other features marking the boundary between the two. This approach is therefore incapable of detecting 'gestalt'[2] events; only that a certain type of event or process is occurring.

2.1 Segmentation by spectral shape

If broad spectral features are used to assess textural similarity, then a segmentation can be obtained where each segment is spectrally homogeneous. This is the approach taken by Aucouturier, Pachet, and Sandler (2005), who use Mel-frequency cepstral coefficients (MFCCs), which are sensitive to broad spectral features whilst remaining relatively invariant to fine spectral (pitch) structure.

Foote (1999) proposed the dissimilarity matrix or S-matrix, which is a matrix of pairwise dissimilarities between all pairs of frames in the signal. Each frame is represented by a vector of MFCC coefficients and the dissimilarity between these tuples measured using a cosine distance measure, which is essentially a function of the angle between two vectors. With the initial analysis at 100 frames per second, this means that a 3-minute song produces an 18000×18000 S-matrix. This extremely large, dense data object is the basis for Foote's methods, which are related to the recurrence plots discussed in Eckmann, Kamphorst, and Ruelle (1987) . For instance, Foote proposed that segments should be defined as 'self-similar' intervals delimited by boundaries corresponding to peaks in a 'novelty' function computed by correlating a Gaussian-tapered 'checkerboard' kernel along the main diagonal of the dissimilarity matrix.

Logan and Chu (2000), also using MFCCs as their spectral features, proposed a method for summarization employing both hidden-Markov models (HMMs) and threshold-based clustering methods, grouping features into key song segments. Peeters, Burthe, and Rodet (2002) propose a multi-pass clustering approach that uses both k-means and HMM-based clustering using multi-scale MFCC features. However, these studies provide no measure of performance for all segments in a song, and (as discussed in Section 1.2) exhibit segment fragmentation.

2.2 Segmentation by harmony

Some recent studies addressed the structure extraction problem in terms of harmonic features rather than timbral texture. For example, Wakefield (1999) proposed chromagram features that represent the distribution of power spectrum energies among the twelve equal-temperament pitch classes of the Western tonal system. This provides invariance to timbral changes such as those observed in repetitions of a melody with different instrumentation.

One desirable property of harmonic features is the possibility of implementing explicit transpositional invariance; this is the property allowing identification of like segments with a constant pitch interval offset: a common device employed in repeated choruses in pop music. Goto (2003) describes a system called *RefraiD* that locates

[2] Think of these as events that are intrinsically countable, e.g.'running a lap' as opposed to 'running' generally. There is discussion of this and related issues in the literature on temporal logics and event calculi (e.g. Allen, 1984; Galton, 1987; Shoham, 1988).

repeated structural segments independent of transposition. The *RefraiD* system is able to track a chorus, for example, even if it modulates up a sequence of semitone key changes. The problem of chorus extraction was divided into four stages: computation of acoustic features and similarity measures; repetition judgement criterion; estimating end-points of repeated sections; and detecting modulated repetitions. The results for chorus detection were reported as accurate for 80 of 100 songs.

Dannenberg and Hu (2002) also describe a system that used agglomerative clustering with chroma-based features for music structure analysis of a small set of Jazz and Classical pieces. They do not report an evaluation of the methods over a corpus, but the figures showing their segmentations display symptoms of fragmentation (see Section 1.2) in some cases.

2.3 Segmentation by rhythm and pitch

Symbolic approaches to structure analysis, such as Orio and Neve (2005), attempt to identify repeated thematic material in string-based representations of music, for instance through statistical analysis of n-grams (Downie & Nelson, 2000). Whilst these methods show promise in identifying structure from score information, they are not well adapted for use in structure analysis from audio, largely due to the addition of significant uncertainty in audio representations; instead, these approaches are primarily used for document retrieval in query by example (or query by humming) systems.

There has recently been some work on combined audio and symbolic representations, attempting to unify the different views of similarity. Maddage et al. (2004) describe a system in which a partial transcription is used to make decisions about structure, integrating beat tracking, rhythm extraction, chord detection and melodic similarity in a heuristic framework for detecting all segments in a song. They also propose using octave-scale rather than Mel-frequency scale cepstral coefficients as pitch-oriented representation. The authors report 100% accuracy for detecting instrumental sections in songs (as against sung sections with instrumental accompaniment), and report results for detection and labelling of *verse*, *chorus*, *bridge*, *intro* and *outro* sections. Similarly, Lu, Wang, and Zhang (2004) describe an HMM-based approach to segmentation that used a $\frac{1}{12}$th-octave constant-Q filterbank for pitch selectivity in addition to MFCC features. They report improved performance in segmentation for the constant-Q transform when used with MFCCs over use of MFCCs alone, when using an S-matrix approach with an exhaustive search to find the best fit segment boundaries to a given objective function.

3 Overview of processing chain

The processing chain we use in our system is motivated by a desire to represent short-term spectral and dynamical structure in a compact way, while being invariant to characteristics or details of the signal which are thought to be less relevant. Thus, we use constant-Q log-power spectra as they approximate some of the features of the human auditory system (Brown, 1991); principal components analysis (PCA) for dimensionality reduction to focus on the spectral features of greatest variance; and hidden Markov models (HMMs) to quantise the space of spectral shapes and capture some of its temporal dynamics. All of this represents a large reduction in the amount of data which makes it feasible to apply the final stage of segmentation using a probabilistic model operating on a time-scale more suitable for the sort of segmentations we wish to achieve. We give further details of this processing chain below.

The first step is to form the signal into a sequence of overlapping frames. Then, a constant-Q or logarithmically-banded spectrogram (Brown, 1991) is computed, covering the range of frequencies from approximately 50 Hz up to the Nyquist frequency at 5.5 kHz with bands 1/12 octave wide. This means that the constant-Q spectogram is sensitive to pitch structure, since the pitch scales of tonal Western music are also logarithmic with 12 equally-spaced tones per octave. We convert the constant-Q spectrum to a logarithmic power scale to model perceived loudness and divide by the sum of the squares of the resulting values. Letting $z_m^{(n)}$ denote the mth band of the nth short-term log-frequency power spectrum in the sequence, with M bands in total, we compute

$$w^{(n)} = \left(\sum_{m=1}^{M} \left(\log z_m^{(n)} \right)^2 \right)^{1/2} \quad \text{and} \quad u_m^{(n)} = \frac{\log z_m^{(n)}}{w^{(n)}}. \tag{1}$$

The M sequences of components $z_m^{(1)}, \ldots, z_m^{(N)}$ computed from one song are collected into an array $\vec{X}$ after subtracting the mean of each band to ensure that each row sums to zero:

$$X_{mn} = u_m^{(n)} - \frac{1}{N} \sum_{n'=1}^{N} u_m^{(n')}. \tag{2}$$

The speech recognition community uses the Discrete Cosine Transform on Mel spectrum coefficients to generate a ‘cepstrum’, which has the empirical property of being approximately decorrelated. Furthermore, higher-order components of the DCT can be discarded, as they typically correspond to low-eigenvalue dimensions. While this transformation is well-motivated (Merhav & Lee, 1993) and not incompatible with some forms of music (Logan, 2000), we choose instead to remove this possible source of bias by using principal components analysis (PCA) to generate our cepstrum, keeping 20 principal components (Aucouturier, Pachet, & Sandler, 2005) of a total of $M = 81$. PCA can be implemented using a singular value decomposition (SVD) of the data matrix, i.e., finding a factorisation $\vec{X} = \vec{U}\vec{S}\vec{V}^T$ where $\vec{U}$ and $\vec{V}$ are orthogonal matrices and $\vec{S}$ is diagonal.

The spectral intensity and shape, represented using the results of PCA combined with the sequence of 2-norms $w^{(1)}, \ldots, w^{(N)}$ defined in (1), are used to construct HMMs with a given number of states and Gaussian observation distributions. For our investigations, we have used HMMs with between 10 and 80 hidden states, though the results presented in this paper were derived from 60-state HMMs only, since this number yielded good results in our previous work on segmentation (Abdallah et al., 2005). Specifically, the data modelled by our HMMs consist of a sequence of 21-component vectors where the nth vector is $(w^{(n)}, V_{n,1}, \ldots, V_{n,20})$ and $\vec{V}$ is the matrix of principal components obtained in the previous step.

The HMMs are not shared between input signals; there is one per song. Each HMM is trained using the Baum-Welch method, after which the most probable state sequence for the trained HMM is determined using the Viterbi algorithm.

Although we use hidden Markov models, we do not use the most likely HMM state occupancy directly as a proposed high-level segmentation, because the dynamics of an HMM are such that it does not favour long, connected segments of a single state. Instead, we use the HMM as a song-specific model of *short-term* spectral dynamics. The HMM states and their output distributions cover and discretise the space of typical spectral shapes for each piece while respecting a degree of temporal coherence between adjacent frames. When we come to modelling segment classes with particular distributions over HMM states, these distributions correspond to distributions over spectral shapes which may be difficult to model directly in the continuous spectral space, using, for example, a small number of Gaussian components. These considerations suggest that we use a relatively high number of hidden states in our models to represent a large number of sound types, rather than restricting our models to a number of states corresponding to the typical number of segments in a song.

Finally, we take the state sequence from the Viterbi algorithm and generate state path histograms using another sliding window, counting the number of each state present in the window.

The motivation behind this histogramming step comes from the histogram clustering method against which we compare our duration-modelling approach, and can be understood by considering the geometry of clustering histograms. The space of short-term histograms over a finite alphabet (in this case, the set of available HMM states) is necessarily discrete, and for very short windows, quite small, as shown in Fig. 3. In the case of few HMM states and few observations per histogram, there simply isn't room in this space for many distinct clusters to form. By using a longer window, we enlarge the space of possible histograms so that a clustering algorithm is more likely to find some interesting structure.

These considerations are less relevant for the duration-aware model we describe in Section 5, since, using a duration prior which penalises short segments, the distribution over HMM states for each segment type (obtained by averaging over instances of that

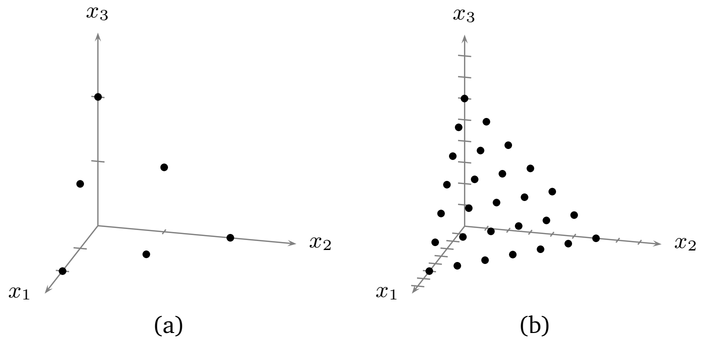

Fig. 3 Histogram spaces for short-term histograms over a domain of 3 HMM states (hence the 3 axes: one for each bin count), for window lengths of (a) 2 and (b) 6. Each point represents a possible complete histogram. Note how, in (a), there is little room for any clustering in the conventional sense, but in (b), there begins to be enough room for well-separated clusters surrounded by low density regions

segment type) is strongly encouraged to reflect runs of neighbouring states. However, we retain the histogramming step in order to make a direct comparison between the two methods on the same sequence of histograms.

4 Segmentation by clustering

Assuming, as we are, an approach based on textural similarity, a commonly used tactic is one which involves three steps:

1. 'atomization', dividing the signal into a number of equal-length fragments at the temporal resolution required for the boundaries and computing the value of the textural property (or feature tuple) for each fragment;
2. clustering the collection of property *values* ignoring the temporal relationships between the fragments to which they belong and thereby assign a class label to each fragment;
3. agglomerating the runs of equally classified fragments into segments.

Since stage (2) produces a discrete-valued sequence, stage (3) is trivial; for example, consider the sequence

$$s = [a, a, a, a, c, c, c, d, d, d, d, b, b, b, c, c]$$

where a, b, c and d are members of a discrete set $\mathcal{C}$. We can treat such a sequence as a function $s : (1..16) \rightarrow \mathcal{C}$, (where $M..N$ denotes the set of integers from M to N inclusive) since there are 16 elements in the sequence. In this case, we can simply place boundaries between every consecutive non-equal pair of elements and identify the segments as the set of maximally long intervals such that the function s takes exactly one value over each interval. In addition, the segments themselves can inherit the classification of their constituent fragments. Overall, the segmentation can be taken to be a set of pairs of type ($\mathsf{interval}(1..20) \times \mathcal{C}$):

$$\{(1..4, a), (5..7, c), (8..11, d), (12..14, b), (15..16, c)\}.$$

However, this algorithm is liable to produce excessively fragmented segments if the clusters identified at stage (2) overlap, since fragments are classified without regard to the classifications of their temporal neighbours. This behaviour can be traced to a failure to imbue the model with our prior expectations about the durations of the segments we wish to detect, a subject to which we return in Section 5.1.

4.1 Pairwise clustering

The histograms resulting from the processing steps of Section 3 inhabit a space which is not self-evidently Euclidean; clustering methods based on Euclidean feature values are therefore not trivially applicable. One way to proceed is to define an empirical dissimilarity measure between observed windowed state histograms with reasonable properties: histograms with the same distribution should be maximally similar, while those with no overlap should be maximally dissimilar.

One such measure is the cosine dissimilarity measure as used by Foote (1999). In Abdallah et al. (2005), we proposed a different dissimilarity measure: a symmetrized Kullback-Leibler (KL) divergence, where we interpret the histograms as summaries of data drawn from a discrete probability distribution. For histograms $\vec{x}$ and $\vec{x}'$ with equal total counts $\sum_i x_i = \sum_i x'_i = N$, we set

$$d_{\mathrm{kl}}(\vec{x}, \vec{x}') = \frac{1}{N} \sum_{i=1}^{M} \left(x_i \log \frac{2x_i}{x_i + x'_i} + x'_i \log \frac{2x'_i}{x_i + x'_i} \right), \qquad (3)$$

This is the sum of the KL divergences from both histograms to their mutual average $(\vec{x} + \vec{x}')/2$, and can be interpreted as measure of the likelihood that the two histograms are actually realisations of the same underlying distribution.

There was no significant difference in performance between the cosine measure and d_{kl} for a segmentation task using the clustering algorithm of Hofmann and Buhmann (1997). However, the probabilistic interpretation of the symmetrized KL divergence suggests the use of other probabilistic clustering models, as explained below.

4.2 Model-based central clustering

An alternative to pairwise clustering is to adopt an explicit generative probabilistic model for the putatively clustered data. This involves a discrete latent variable model,[3] where, for each time $i \in 1..L$, there is an unobserved discrete class label $c_i \in 1..K$, where there are K possible classes. These classes will model K segment classes. We then assume that each class defines a conditional probability density $p_{X|C}$ for the data belonging to that class. The pdf of the data as a whole will be a weighted mixture of these class-conditional densities.

Since the data we wish to cluster are histograms representing a distribution over a discrete feature space (the HMM states), we may, following Puzicha, Hofmann, and Buhmann (1999), consider each underlying class to determine a probability distribution over the feature space. The observed histograms are then modelled as the result of drawing samples from one of these distributions. This leads quite naturally to a probabilistic latent variable model with an optimisable likelihood function.

Assuming the existence of K underlying classes, and a mixture model parameterised by θ, the joint probability of observing the jth HMM state from the kth class is

$$p(j, k \mid \theta) = p_{X|C}(j \mid k, \theta) p_C(k \mid \theta), \qquad (4)$$

where $p_C(k \mid \theta)$ is the weighting of the kth mixture component in the model associated with the parameter θ. In our experiments we fixed $p_C(k \mid \theta) = 1/K$ for all $k \in 1..K$. The discrete class-conditional distributions were parameterised by an $M \times K$ matrix $\vec{A}$, such that $p_{X|C}(j \mid k, \theta) = A_{jk}$. Since $p_C(k \mid \theta)$ was not dependent on either k or θ, $\vec{A}$ was the only parameter required, so we could set $\theta = \vec{A}$.

If, instead of observing a single HMM state, we observe a sequence of such states which we then summarise as a histogram $\vec{x}$ such that x_j is the number of observations

[3] That is, one where we introduce new variables in addition to those representing the observed data.

of the jth state, then the joint probability of the pair $(\vec{x}, k)$ (assuming that all the observations are associated with a single realization of the class variable k) is

$$p(\vec{x}, k \mid \theta) = W(\vec{x}) \left(\prod_{j=1}^{M} \left[p_{X|C}(j \mid k, \theta) \right]^{x_j} \right) p_C(k \mid \theta), \tag{5}$$

where $W(\vec{x})$ is the multiplicity factor for the histogram $\vec{x}$ and accounts for the number of distinct ways in which that histogram could have been realised.

Given a *sequence* of L such histograms encoded as an array $\vec{X} \in \mathbb{N}^{M \times L}$, where the ith column of $\vec{X}$ is a histogram over M possible HMM states, and a corresponding sequence of class assignments $\vec{c} \in (1..K)^L$, the overall log-probability of the pair $(\vec{X}, \vec{c})$ with respect to the model parameterised by $\theta = \vec{A}$, with $p_C(k \mid \theta) = 1/K$, is

$$\log p(\vec{X}, \vec{c} \mid \theta) = \sum_{i=1}^{L} \left(\log W(X_{:i}) - \log K + \sum_{j=1}^{M} X_{ji} \log A_{jc_i} \right) \tag{6}$$

where $X_{:i}$ denotes the ith column of the array $\vec{X}$. If we assume that each histogram contains the same number of observations J, then Stirling's approximation yields

$$\log p(\vec{X}, \vec{c} \mid \theta) \approx \sum_{i=1}^{L} \left(\sum_{j=1}^{M} X_{ji} \log \frac{A_{jc_i}}{X_{ji}} + J \log J - \log K \right) \tag{7}$$

For the purposes of constructing an energy function, we can disregard any additive terms independent of $\vec{c}$ and θ. Hence, the following energy function is sufficient to specify the model exactly [4]:

$$\varepsilon(\vec{c}, \theta) = \sum_{i=1}^{L} \sum_{j=1}^{M} \sum_{k=1}^{K} \delta(k, c_i) X_{ji} \log \frac{X_{ji}}{A_{jk}}. \tag{8}$$

The model's parameters are optimised using a Deterministically-Annealed EM (DAEM) algorithm as described by Puzicha, Hofmann, and Buhmann (1999); the end result is a maximum *a posteriori* estimate for the class assignments $\vec{c}$ and the class-conditional distributions $\vec{A}$.

5 Modelling temporal coherence

When applied to the problem of audio segmentation, the clustering algorithms described in Section 4 often result in more fragmented segments than we would like to see. This is partly because such methods do not model any expectation of temporal

[4] Note that the use of Stirling's approximation for the multiplicity factor does not affect the validity of the energy function, since both the multiplicity and its approximation are functions of $\vec{X}$ alone, and hence constants in any given problem.

coherence which we may intuitively bring to the problem. In essence, the clustering methods associate audio fragments only on the basis of their similarity in some feature space, and are blind to the notion of association by temporal proximity: as humans, we are prepared to accept that otherwise dissimilar fragments are more likely to belong to the same segment if they are adjacent in time than if they are widely separated. One way to incorporate this into the model is to introduce an explicit prior on the duration of the segments found.

First order hidden Markov models have (discrete) exponential state duration distributions, the mean duration being a function of the self-transition probability. It is far from evident that this exponential distribution is appropriate for musical segments, and indeed this is the reason that there is further processing after the derivation of the most probable state sequence: if the HMM had the correct dynamics, further processing might be unnecessary. For instance, one can imagine using HMMs with special state structures: if several states form a chain (i.e. state a can only be followed by b, which can only be followed by c etc.) then the duration of the composite pattern will be the sum of several exponentially distributed durations, which will have a Gamma distribution with two degrees of freedom per distinct state in the chain.

Alternatively, we can implement HMMs with explicit duration distributions, as in Rabiner (1989) for example. However, the computational cost of this is non-negligible, so we leave investigation of this area for future work, and instead discuss processing steps which can explicitly model temporal coherence in the output state sequence from an ordinary HMM, which we interpret as modelling the low-level dynamics of the audio-generating processes.

5.1 Duration distributions

We describe below some families of pdfs that could be used as prior segment duration distributions. We have removed any scale parameter since one can be added to any univariate density in a mechanical way. In any case, we must also consider how a continuous-time duration prior maps to a discrete-time duration prior given the relationship between continuous time and discrete time induced by the discretisation of the original signal and the two subsequent windowing operations associated with computing the short-term constant-Q spectra and the short-term histograms of HMM state occupancy.

5.1.1 Gamma distribution $\mathcal{G}(\alpha)$

The Gamma random variable we describe here is a restriction of the usual two parameter Gamma distribution where the scale parameter has been fixed at 1. If $X \sim \mathcal{G}(\alpha)$, then

$$p_{\mathcal{G}}(x \mid \alpha) = \frac{x^{\alpha-1}e^{-x}}{\Gamma(\alpha)}. \tag{9}$$

This can be derived from an 'energy' function $\varepsilon_{\mathcal{G}}(x, \alpha) = x - (\alpha - 1)\log x$, such that $p_{\mathcal{G}}(x \mid \alpha) = e^{-\varepsilon_{\mathcal{G}}(x,\alpha)}/\mathcal{Z}_{\mathcal{G}}(\alpha)$, where the normalisation factor is

$\mathcal{Z}_{\mathcal{G}}(\alpha) = \int_0^\infty e^{-\varepsilon_{\mathcal{G}}(x,\alpha)}\mathrm{d}x$. The Gamma distribution is not 'scale free', and tends to a Gaussian as $\alpha \to \infty$.

5.1.2 *Inverse Gamma distribution $\mathcal{IG}(\gamma)$*

An inverse Gamma random variable is the reciprocal of a Gamma random variable: $X \sim \mathcal{IG}(\gamma)$ if $X^{-1} \sim \mathcal{G}(\gamma)$. If $X \sim \mathcal{IG}(\gamma)$, then

$$p_{\mathcal{IG}}(x \mid \gamma) = \frac{e^{-1/x}}{x^{\gamma+1}\Gamma(\gamma)}. \tag{10}$$

This can be derived from an energy function $\varepsilon_{\mathcal{IG}}(x,\gamma) = 1/x + (\gamma+1)\log x$. Because of the power-law tail of the distribution, the inverse Gamma is 'scale-free' down to a lower cut-off determined by γ.

5.1.3 *An alternative non-negative random variable*

The parameter α of the Gamma distribution mainly affects the behaviour of its pdf around zero, whereas the parameter γ of the inverse Gamma distribution mainly affects the shape of its tail. For our segmentation model, we wish to have available a duration distribution with the scale-free power-law tail of the inverse Gamma pdf, representing our general ignorance about the duration of segments, but with a controllable cut-off for segments below a certain duration. With this in mind, we consider the pdf defined by the energy function

$$\varepsilon_{\mathcal{H}}(x,\nu,\gamma) = \frac{1}{\mid \nu \mid}x^{-\nu} + (\gamma+1)\log x. \tag{11}$$

This includes both the inverse Gamma family (obtained by setting $\nu = 1$), and the Gamma family (obtained by setting $\nu = -1$). The parameter ν allows very short segments to be suppressed to a greater or lesser degree. In the experiments we present in this paper, after examining the empirical distribution of the human-annotated segments, we set $\nu = 2$ and $\gamma = 0$. With $\nu \geq 0$ and $\gamma = 0$, the mode of the distribution is always 1 independent of ν.[5] We also applied a scale factor such that the mode was at 20 s. The values of ν and the scale factor were chosen to match approximately the distribution of segment durations in the ground-truth segmentations.

5.2 Computing an interval-based representation

In the histogram clustering model of Section 4.2, we represent the latent structure as a sequence of class assignments $\vec{c} \in \mathcal{C}^L$ defined on a regularly sampled discrete timeline of length L. The final segmentation is performed by agglomerating runs of similarly classified timepoints. In order to use a duration prior during the optimization itself, this segmentation process must be brought into the computation of the cost function

[5] Note that $\gamma = 0$ implies an improper prior for the segment length; when applied to a given problem of finite length, however, the prior is necessarily normalizeable. See Appendix B for more details.

to be optimised, rather than applied post-hoc after the optimisation is complete as we did in the segmentation algorithm based on clustering.

Thus, we define a function $\mathsf{segments} : \mathcal{C}^* \to (\mathit{interval}(\mathbb{N}) \times \mathcal{C})^*$, which returns a list of classified segments. (The notation A^* denotes the type of a sequence of indefinite length consisting of elements of type A.) We also define the functions $\mathsf{fst} : (A \times B) \to A$ (for any two types A and B) such that $\mathsf{fst}(x, y) = x$, and $\mathsf{dur} : \mathit{interval}(\mathbb{N}) \to \mathbb{N}$ which gives the duration of a discrete-time interval. In these terms we can define a duration-aware distribution for a random variable $C^* : \mathit{rv}(\mathcal{C}^*)$, of which $\vec{c}$ is a realization. Letting $p_D : \mathit{pdf}(\mathbb{N})$ be a probability density over discrete-time durations, we set

$$p_{C^*}(\vec{c}) = \prod_{i=1}^{\mathsf{len}(\sigma)} p_D(\mathsf{dur}(\mathsf{fst}(\sigma_i))) \text{ where } \sigma = \mathsf{segments}\,(\vec{c}), \tag{12}$$

that is, the probability of a segmentation is the product of the probabilities of the durations of each of its segments.

5.3 Constructing a discrete-time duration distribution

The discrete-time duration pdf p_D is defined in terms of the desired continuous time duration distribution and an object M which embodies the relationship between continuous physical time and the discrete timeline of the model. This relationship is defined by the initial discretisation of time implied by the sampling of the original audio signal and the subsequent layers of windowing which go into the computation of the constant-Q spectrogram and the HMM state-occupancy histograms, which means that points on the state histogram timeline correspond with overlapping intervals on the continuous timeline. Given this structure, a number of distinct mappings from discrete time intervals and durations to continuous time intervals and durations can be defined, and therefore a discrete time interval could reasonably be associated with one of a number of continuous time durations. If we let $\mathsf{fd}_M : \mathbb{N} \to \mathbb{R}$ denote one such function from discrete time durations to continuous time durations, then to use, for example, the inverse Gamma duration model $\mathcal{IG}(\gamma)$, we would set

$$p_D(d) = \frac{p_{\mathcal{IG}}(\tau^{-1}\mathsf{fd}_M(d) \mid \gamma)}{\sum_{l=1}^{L} p_{\mathcal{IG}}(\tau^{-1}\mathsf{fd}_M(l) \mid \gamma)}, \tag{13}$$

where τ is a scale factor. The limitation on segment duration in the sum in the denominator of (13) is required to obtain a properly normalised distribution; using the length L of the discrete timeline would seem to be an uncontroversial choice since no interval can be longer than this.

5.4 Statistical model including durations

The histogram clustering model can be augmented with a segment duration model by removing the flat prior $p_C(k \mid \theta)$ in (5) and subtracting from (8) the logarithm of the duration-aware prior $p_{C^*} : \mathit{pdf}(\mathcal{C}^*)$, defining $d_l(\vec{c})$ as the duration of the lth segment implied by the sequence of class assignments $\vec{c}$ and neglecting constant terms:

$$\begin{aligned}\varepsilon^*(\vec{c},\theta) &= \varepsilon(\vec{c},\theta) - \log p_{C^*}(\vec{c}).\\ &= \sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{k=1}^{K}\delta(k,c_i)X_{ji}\log\frac{X_{ji}}{A_{jk}}\\ &\quad+\sum_{l}\varepsilon_{\mathcal{H}}(\tau^{-1}\mathsf{fd}_M(d_l(\vec{c})),\nu,\gamma)\end{aligned} \tag{14}$$

In our experiments, the parameters of the duration prior were treated as given, not subject to optimization as part of the algorithm, and therefore need not be incorporated into the parameter θ.

We have described the segmentation model using the histogram observation model used in the histogram clustering method, but it can just as easily be applied to any observation model for which we can compute and optimise likelihoods, such as a Gaussian mixture model.

The duration prior introduces a strong coupling between the class variables at nearby sites on the discrete timeline, which makes the posterior density $p(\vec{c} \mid \vec{X},\theta)$ rather difficult to work with as part of an EM, or DAEM (deterministically annealed EM) algorithm. Although it may be possible to construct a good variational approximation to this posterior by exploring alternative parameterizations, our initial approach has been direct Markov chain Monte Carlo (MCMC) simulation of the posterior, which replaces the exact E-step. The statistics collected during the simulation are used in the M-step.

6 Inference methods

6.1 Markov-Chain Monte Carlo algorithms

The following MCMC algorithms all fit into the general framework of a Metropolis-Hastings sampler, but with different proposal distributions. See, e.g., Robert & Casella (1999) for a general introduction to MCMC methods.

6.1.1 Gibbs sampling

A Gibbs sampler (Robert & Casella, 1999) is a Metropolis-Hastings algorithm in which the proposed steps involve a change in just one component of a multi-component state variable, chosen in such a way as to ensure that the probability of accepting the step is 1. In our case, the state variable is the class-assignment sequence $\vec{c} : \mathcal{C}^L$, and each step involves resampling one of the class assignments, e.g.the ith one c_i, from the conditional distribution $p(c_i \mid c_{1..L\setminus i}, \vec{X}, \theta)$, where $c_{1..L\setminus i}$ denotes the sequence of class assignments with the ith element removed. The site i to be updated at each iteration can be chosen at random from a uniform distribution on $1..L$ (as in Algorithm 1, describing one step of the Gibbs sampler) or in some some deterministic pattern that visits all sites equally often. This prescription ensures that no other steps are

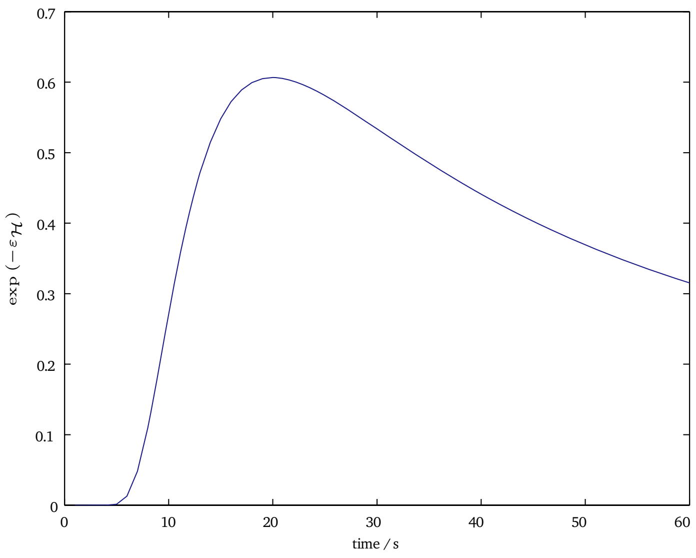

Fig. 4 The prior used for segment durations

needed to ensure detailed balance. In Algorithm 1 and henceforth, we denote the target probability distribution over class assignment sequences as $p_t : \mathsf{pdf}(\mathcal{C}^L)$, and the sequence obtained by assigning sites $i..j$ in the sequence $\vec{c}$ to class k as $\vec{c}_{i..j}^{\leftarrow k}$. In the following, $\mathcal{U}(1..L)$ denotes a random variable distributed uniformly over the integers 1 to L.

Algorithm 1 $\mathsf{MC}_{\mathsf{Gibbs}} : (\vec{c}, p_t) \mapsto \vec{c}'$

```
$i \leftarrow$ sample from $\mathcal{U}(1..L)$
$k \leftarrow$ sample from pdf $p(k \mid c_{1..L\setminus i}, \vec{X}, \theta) \propto p_t(\vec{c}_{i..i}^{\leftarrow k})$
$\vec{c}' \leftarrow \vec{c}_{i..i}^{\leftarrow k}$
```

Unfortunately, while the Gibbs algorithm ensures that every proposed step is taken, it also means that, once any contiguous segments have formed, almost all proposed steps involve no change of state. This is because a duration prior of the sort we are interested in (such as that in Fig. 4), favouring longer segments and penalising very short ones, will strongly inhibit the formation of the very short segments that would result from changing the classification of single sites embedded in longer segments. It also means that the algorithm has trouble eliminating short incorrectly classified domains. The result is that, with a strong duration prior, we would have to sample for a long time to explore the space of probable configurations and therefore to have any confidence in the EM optimisation as a whole; the 'wrong' segmentations get frozen in and cannot be rectified.

The same phenomenon, known as 'critical slowing down', is observed in Gibbs or Metropolis simulations of spin systems (e.g. the Ising or Potts models) as the temperature approaches the 'critical temperature' at which the correlation length

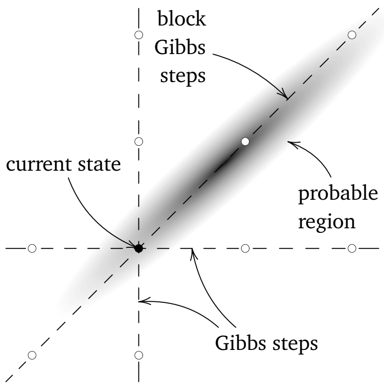

Fig. 5 A Gibbs sampler can get stuck when none of the steps available to it lead to configurations of significant probability. Even if other probable configurations exist, the Gibbs sampler is unlikely to reach them. However, the states can sometimes be reached by updating several sites at once, as in the block-Gibbs sampler. (The two-dimensional space of the figure represents the usually much higher-dimensional, state space of the system in question)

diverges—this is the point of phase transition where, as the temperature decreases, formation of large similarly-classified domains becomes favoured.

In order to address this problem, we turn for inspiration to the methods adopted by statistical physicists in simulating Ising and Potts systems near their critical temperatures; similar approaches have been taken in the image segmentation community (see Barbu & Zhu (2004) for one application, similar to that described below, of the method of Swendsen & Wang (1987)).

6.1.2 The block-Gibbs sampler

In the state space of the system, the region of significant probability density is highly constricted by the introduction of a duration prior. If the current state is a relatively probable one, most single-site updates correspond to steps that would take the system out of the high probability region (because they correspond with the introduction of very short segments) and so are not often generated by the single-site Gibbs sampler (see Fig. 5).

In a block-Gibbs sampler, we update several adjacent sites all at once instead of single sites one at a time. This can lead the system in a single jump to a probable configuration that would have taken a Gibbs sampler a very long time to reach updating one site at a time.

To implement a block-Gibbs sampler, one must decide on a procedure for selecting which block is going to be sampled at each iteration. In the next section, we investigate a variant of the Wolff algorithm for selecting these blocks.

6.1.3 A Wolff-Gibbs algorithm

The Wolff algorithm (Wolff, 1989) was designed to simulate Ising, Potts and $x - y$ systems near their critical temperatures while avoiding the critical slowing down phenomenon found in single-site update algorithms like Gibbs sampling and the Metropolis algorithm. It can be thought of as a block-update sampler where choice of which block of sites to consider at each step is carefully tuned to the temperature of the system and its current configuration in such a way that the proposed steps are always accepted.

Algorithm 2 $\mathsf{MC}^{\alpha}_{\mathsf{WG}} : (\vec{c}, p_t) \mapsto \vec{c}'$
(Note: the random variable $\mathcal{B}(\alpha)$ is 1 with probability α and 0 otherwise.)

```
Procedure choosedomain : $(\vec{c}, \alpha) \mapsto (i, j)$
$l \leftarrow$ sample from $\mathcal{U}(1..L)$
$i \leftarrow l; j \leftarrow l$
while $i \neq 1 \wedge c_{i-1} = c_i \wedge$ (sample from $\mathcal{B}(\alpha) = 0$) do
  $i \leftarrow i - 1$
end while
while $j \neq L \wedge c_j = c_{j+1} \wedge$ (sample from $\mathcal{B}(\alpha) = 0$) do
  $j \leftarrow j + 1$
end while

Function $b : (\vec{c}, i..j, k') \mapsto n$
  $n = \mathcal{I}(i = 1 \vee c_{i-1} \neq k') + \mathcal{I}(j = L \vee c_{j+1} \neq k')$
  where $\mathcal{I}(\mathsf{false}) = 0,\ \mathcal{I}(\mathsf{true}) = 1$

$(i, j) \leftarrow$ choosedomain $(\vec{c}, \alpha)$
$k \leftarrow$ sample from pdf $p(k) \propto p_t\left(\vec{c}^{\leftarrow k}_{i..j}\right)\alpha^{-b(\vec{c},i..j,k)}$
$\vec{c}' \leftarrow \vec{c}^{\leftarrow k}_{i..j}$
```

The procedure is to choose an initial site as in the Gibbs sampler, but then to expand this into a block of contiguous sites by adding adjacent similarly classified sites contingent on a random test. This is known in the literature as ‘cluster’ growing, which is a little unfortunate here since these have nothing to do with the clustering problem described earlier. To avoid confusion, we will use the term ‘domain’ instead.

Our adapted Wolff-Gibbs algorithm (see algorithm 2 for one Monte Carlo step) involves a Wolff-like domain growing phase followed by a block-Gibbs sampling step which updates the entire domain. A random seed site is chosen and a domain grown from it (leftwards and rightwards), stopping either when a boundary is reached or with probability α at each step. These sites are then reclassified *en masse* with a label chosen using a sampling distribution proportional to the target distribution, with an extra factor of α for each boundary that the reclassified region has; the function b counts these boundaries. These extra factors are required in order to preserve detailed balance as we show in the next section.

6.1.4 Detailed balance in the Wolff-Gibbs algorithm

In the original Wolff algorithm, the conditions for domain growth ensure that once the domain is chosen, neither a Gibbs-like sampling step nor an acceptance test are required to ensure detailed balance. In our version, with an arbitrary duration prior and a Gibbs sampling step, we must adjust the sampling distribution to satisfy the condition of detailed balance; this is the purpose of the extra factor of $\alpha^{-b(\vec{c},i..j,k)}$ in the sampling distribution of Algorithm 2. In this section we show how this correction is derived.

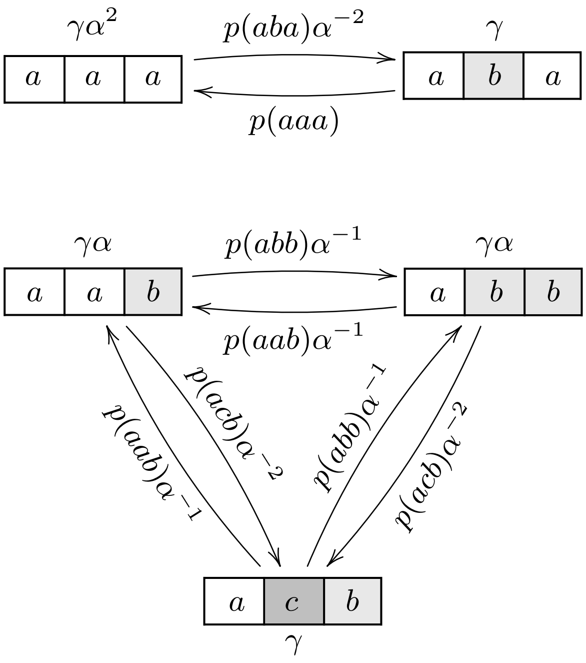

Fig. 6 Two 'cliques' of configurations that are linked by reassignment of the central interval. Each a, b, or c represents a *sequence* of sites classified a a, b, or c, so each arrow represents a *block* assignment of the central section. The expressions above and below the boxes show the relative probabilities of proposing the central section in each case; the common factor is $\gamma = (l/L)(1-\alpha)^{(l-1)}$ (see Eq. (15)). The arrows are labelled by the relative probability of the step according to the sampling distribution. These five cases include all possible local configurations for the central domain within the context of the two neighbouring segments, up to permutations of the labels

In the following, we will need to refer to members of the set of configurations obtained from a base configuration $\vec{c} \in \mathcal{C}^L$ by reassigning the sites in the interval $i..j$. We can think of this set of configurations $\vec{c}_{i..j}^{\leftarrow k}$ for possible class assignments k as a 'clique', in the sense that each one can be reached by a block reassignment from all the others (see Fig. 6). Note that, for $i..j$ to be a valid domain proposition, the configuration $\vec{c}$ must itself be a member of the clique, with $\vec{c} = \vec{c}_{i..j}^{\leftarrow k}$ (see choosedomain in Algorithm 2) for some k.

The domain selection phase of the Wolff algorithm implies that the probability of choosing the domain $i..j$ given an initial configuration $\vec{c} = \vec{c}_{i..j}^{\leftarrow k}$ is

$$p_{\mathrm{d}}\big(i..j \mid \vec{c}_{i..j}^{\leftarrow k}\big) = \tfrac{l}{L}(1-\alpha)^{l-1}\alpha^{2-b(\vec{c},i..j,k)}, \tag{15}$$

where $l = j - i + 1$, the number of sites in the domain, and b is the same boundary counting function as defined in Algorithm 2. Let the subsequent probability of assigning this domain the class k' be denoted by $r(k' \mid k, i..j, \vec{c})$. The overall probability of moving from $\vec{c}_{i..j}^{\leftarrow k}$ to $\vec{c}_{i..j}^{\leftarrow k'}$ is then

$$T(k' \mid k, i..j, \vec{c}) = r(k' \mid k, i..j, \vec{c})\,p_{\mathrm{d}}\big(i..j \mid \vec{c}_{i..j}^{\leftarrow k}\big). \tag{16}$$

We wish to find a definition of r that fulfils the condition of detailed balance for the target density $p_t : \mathsf{pdf}(\mathcal{C}^L)$, for which we require

$$T(k' \mid k, i..j, \vec{c}) p_t\big(\vec{c}_{i..j}^{\leftarrow k}\big) = T(k \mid k', i..j, \vec{c}) p_t\big(\vec{c}_{i..j}^{\leftarrow k'}\big). \tag{17}$$

Combining these, we find that

$$\frac{r(k' \mid k, i..j, \vec{c})}{r(k \mid k', i..j, \vec{c})} = \frac{p_t\big(\vec{c}_{i..j}^{\leftarrow k'}\big)\alpha^{2-b(\vec{c},i..j,k')}}{p_t\big(\vec{c}_{i..j}^{\leftarrow k}\big)\alpha^{2-b(\vec{c},i..j,k)}}. \tag{18}$$

Bearing in mind that r must also be a properly normalised sampling distribution, the above condition is satisfied by

$$r(k' \mid k'', i..j, \vec{c}) = \frac{p_t\big(\vec{c}_{i..j}^{\leftarrow k'}\big)\alpha^{-b(\vec{c},i..j,k')}}{\sum_{k\in\mathcal{C}} p_t\big(\vec{c}_{i..j}^{\leftarrow k}\big)\alpha^{-b(\vec{c},i..j,k)}}, \tag{19}$$

which reduces to the same distribution as used in Algorithm 2. Note that $r(k' \mid k'', i..j, \vec{c})$ does not depend on k'', and hence the sampling distribution over the clique is the same regardless of which member of the clique is the current state.

6.2 EM with a stochastic E-step and annealing

While MCMC methods allow us to sample from the posterior distribution, we still have a number of options for how we measure the necessary statistics for a stochastic EM algorithm. These range from collecting an entirely new sequence of samples for each iteration of the EM algorithm (providing for maximum independence of samples), to computing a small number of samples for each iteration and using a gradually-decaying memory to compute average sufficient statistics.

In our experiments, we used an exponentially decaying memory which accumulates a short-term approximation to the posterior distribution over configurations:

$$\overline{q}_t = (1-\eta)\overline{q}_{t-1} + \eta\,\mathsf{bmap}(\vec{c}_t), \tag{20}$$

where η is the ‘forgetting’ rate, and $\mathsf{bmap} : (1..K)^L \to \{0, 1\}^{K\times L}$ is a function which converts a discrete valued sequence into a bitmap: if $y = \mathsf{bmap}(x)$, then $y_{ij} = (1 \text{ if } x_j = i; 0 \text{ otherwise})$. We then treat $\overline{q}_t$ as the result of the E-step, and use it as the approximation to the posterior in the tth iteration of an EM algorithm.

We anneal by using a temperature dependent posterior $p_\beta(\vec{c} \mid \vec{X}, \theta)$ as the target distribution and linking the domain growth parameter α to the inverse temperature β by setting $\alpha = e^{-\beta J}$ for some constant J. This is an integral part of the original Wolff algorithm and means that, as the temperature drops (i.e., as $\beta \to \infty$), the propensity to propose large domains increases (since $\alpha \to 0$). This reduces the amount of time spent considering configurations containing short segments which are very unlikely at low temperatures.

7 Experiments

7.1 Test data

The musical test data we used were fourteen tracks of popular music from Sony's catalogue, which were downsampled to 11 kHz mono before being distributed to the MPEG-7 working group. Notional ground truth segmentations and annotations made by expert listeners were provided, giving a start time, end time and textual label for each segment.

For each song, two constant-Q spectrograms were computed using frames 200 ms and 400 ms, with a 50% overlap in all cases, that is, with hop sizes of 100 ms and 200 ms respectively. The frequency resolution was set to $\frac{1}{12}$-octave. The spectrograms were subjected to normalisation and dimension reduction as described in Section 3, retaining the first 20 principal components. A separate 60-state HMM was trained on each song encoded in this way, and the most probable state-path computed by Viterbi decoding. State occupancy histograms were computed using windows of 10 states with a hop size of 5 for the smaller audio frames of 200 ms, and 4 states with a hop size of 2 for the larger 400 ms frames.

For each of the resulting sequences of histograms, we applied the histogram clustering method (Section 4.2) and the duration-aware model (Section 5) with Wolff-Gibbs sampling (Section 6.1.3). In the latter case, the duration prior (11) of Section 5.1.3 was used with the parameters $\gamma = 0$ and $\nu = 2$, with the overall time-scale of the duration distribution determined by setting $\tau = 20$ s in (14), which puts the mode of the duration distribution at 20 s.

7.2 Evaluation methods

To compare the automatically-generated segmentation with the ground truth, it is necessary to map the boundaries between segments back to the original continuous timeline in terms of which the ground truth annotations are given. Bearing in mind that the sequence of short-term histograms is defined on a discrete timeline which is itself derived via two windowing operations from the original discrete time signal, this is not a trivial operation. The boundary between two segments (essentially the 'gap' between two discrete time moments) could conceivably be mapped back to one of several points or intervals on the continuous timeline, depending on the desired semantics of segment classification; consider, for example, the difference between 'this segment contains piano sounds' vs. 'this segment contains only piano sounds'. We shall, for the time being, map the gap between two discrete moments back to the middle of the overlap between their respective continuous time intervals, noting that successive state histograms overlap by approximately 0.5 s on the original continuous timeline.[6]

[6] Compare with the 3.2 s overlap in Abdallah et al. (2005).

7.2.1 Interval matching

Having found times for the detected segment boundaries, we adapt the segmentation evaluation measure of Huang and Dom (1995). Considering the measurement M as a sequence of segments S_M^i, and the ground truth G likewise as segments S_G^j, we compute a directional Hamming distance d_{GM} by finding for each S_M^i the segment S_G^j with the maximum overlap, and then summing the difference,

$$d_{GM} = \sum_{S_M^i} \sum_{S_G^k \neq S_G^j} \left| S_M^i \cap S_G^k \right| \tag{21}$$

where $|\cdot|$ denotes the duration of a segment. We normalise d_{GM} by the track length L to give a measure of the missed boundaries $m = d_{GM}/L$. Similarly, we compute d_{MG}, the inverse directional Hamming distance, and a similar normalised measure $f = d_{MG}/L$ of the segment fragmentation. Note that these measures consider only the time intervals occupied by each segment, not the classifications of the segments.

7.2.2 Information-theoretic label matching

An alternative information-theoretic measure was also investigated in order to assess how well the classification reflected the original segment labels. This involves 'rendering' the ground-truth segmentation into a discrete time sequence of numeric labels $\vec{c}_0 : (1..L) \to (1..K_0)$, using the same discrete timebase as the sequence to be assessed, $\vec{c}_1 : (1..L) \to (1..K_1)$, and then treating the joint distribution over labels as a probability distribution. The two sequences are compared by computing the conditional 'entropies' $H(\vec{c}_1 \mid \vec{c}_0)$ and $H(\vec{c}_0 \mid \vec{c}_1)$ as follows:

$$H(\vec{c}_1 \mid \vec{c}_0) = \sum_{(j,k) \in \mathcal{R}(\vec{c}_0) \times \mathcal{R}(\vec{c}_1)} -p_{01}(j,k) \log_2 \frac{p_{01}(j,k)}{p_0(j)}, \tag{22}$$

$$H(\vec{c}_0 \mid \vec{c}_1) = \sum_{(j,k) \in \mathcal{R}(\vec{c}_0) \times \mathcal{R}(\vec{c}_1)} -p_{01}(j,k) \log_2 \frac{p_{01}(j,k)}{p_1(k)}, \tag{23}$$

$$I(\vec{c}_0, \vec{c}_1) = \sum_{(j,k) \in \mathcal{R}(\vec{c}_0) \times \mathcal{R}(\vec{c}_1)} p_{01}(j,k) \log_2 \frac{p_{01}(j,k)}{p_0(j) p_1(k)}, \tag{24}$$

where $\mathcal{R}(\vec{c}) = \{k \mid \exists i.\vec{c}(i) = k\}$, the set of distinct values in $\vec{c}$, and p_{01} is the normalised joint 2-D histogram of corresponding elements of $\vec{c}_0$ and $\vec{c}_1$; i.e.

$$\forall (j,k) \in \mathcal{R}(\vec{c}_0) \times \mathcal{R}(\vec{c}_1), \quad p_{01}(j,k) = \frac{1}{L} \sum_{i=1}^{L} \delta(j, \vec{c}_0(i)) \delta(k, \vec{c}_1(i)), \tag{25}$$

where L is the length of the sequences $\vec{c}_0$ and $\vec{c}_1$. Correspondingly, p_0 and p_1 are the marginal distributions defined as $p_0(j) = \sum_k p_{01}(j,k)$ and $p_1(k) = \sum_j p_{01}(j,k)$. The $H(\vec{c}_0 \mid \vec{c}_1)$ measures the amount of ground-truth information 'missing' from the class assignments, while $H(\vec{c}_1 \mid \vec{c}_0)$ measures the amount of 'spurious' information in

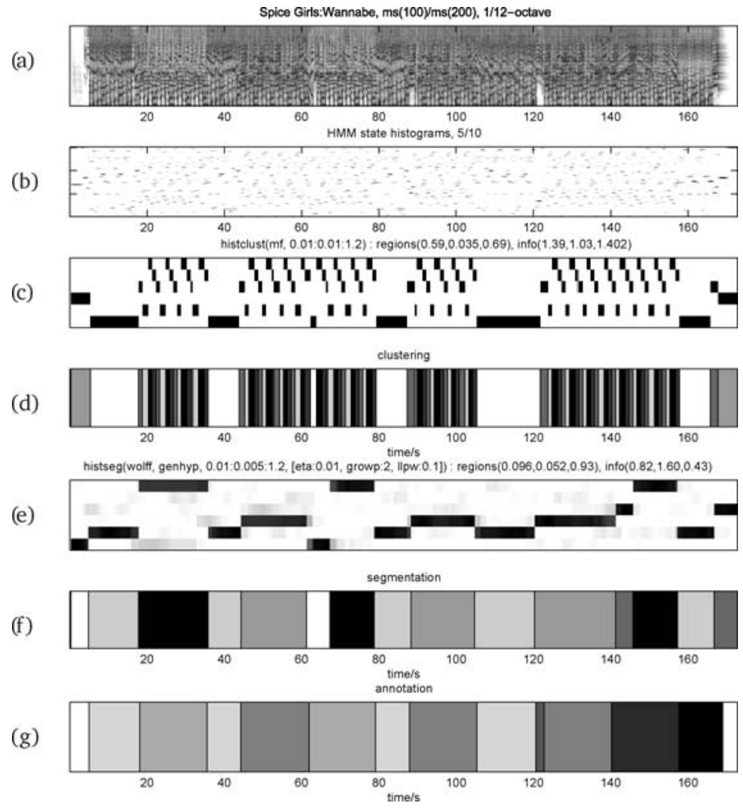

Fig. 7 A segmentation of a song from the test set, comparing the results of central clustering and those of the Wolff-Gibbs segmenting algorithm, both with 6 segment types. The constant-Q spectrogram is displayed in the panel (a), while the 'ground truth' annotations from the expert listener are in panel (g). The results of the Viterbi algorithm are in (b); panels (c) and (d) display two visualisations of the central clustering, while (e) and (f) show the results of the Wolff segmenter. The shades of gray in the lower two panels encode the segment labels, and need only match up to a permutation to signify a perfect segmentation

the classification, e.g. when several classes represent one segment type. The 'mutual information' $I(\vec{c}_0, \vec{c}_1)$ measures the information in the class assignments about the ground truth segment label, and is maximal when each segment type maps to one and only one class. In this case both $H(\vec{c}_1 \mid \vec{c}_0)$ and $H(\vec{c}_0 \mid \vec{c}_1)$ will be zero.

7.3 Results

The two segmentation methods as applied to two of the tracks in our test set are illustrated in Figs. 7 and 8. Note in those figures the results of the central clustering (in panels (c) and (d)), which show fragmentation due to picking out the internal structure of musically coherent segments. The problem is not that the segmentation is manifestly wrong, as the features the central clustering picks out are present in the sound, but that they are not on the timescale of direct interest: they correspond to chords or similar low-level features of the audio. Figure 9 shows an example of the relative

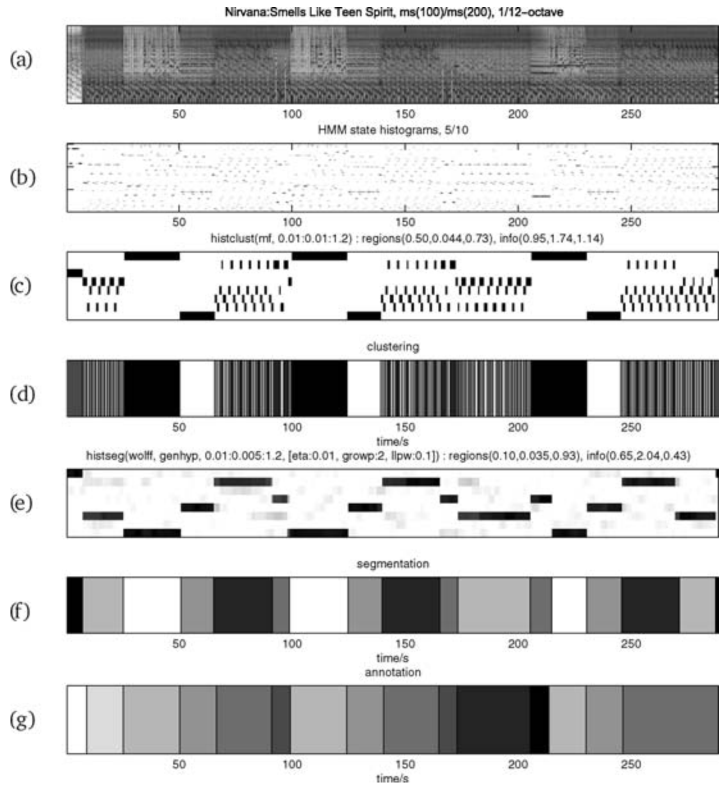

Fig. 8 A segmentation of a different song from the test set comparing the results from the two segmentation methods

insensitivity of our Wolff-Gibbs algorithm to the sizes of the framing windows in the preprocessing stages of Section 3: the results are of similar quality to those of Fig. 8.

To evaluate the performance of our algorithms on our test corpus with the interval matching measure of Section 7.2.1, we plot $1 - f$ against $1 - m$, by analogy with the precision-recall graph used in the field of Information Retrieval. Precision characterises the ability of the system to avoid false positives, analogous to extra segments in our case, and so a decreasing function of f. Recall is the measure of successfully finding relevant matches, analogous to avoiding missed boundaries, and so a decreasing function of m. Thus, in Figs. 10 and 11, the segmentations most closely resembling the ground truth annotations are represented by points close to (1, 1); points near the horizontal axis have large f, suffering from excessive fragmentation relative to the ground truth.

Figure 10 demonstrates the range of quality of segmentations generated by the histogram clustering method. Note in particular that $1 - f$ is strongly weighted towards small values, indicating that there is significant fragmentation in the generated segmentations (as can be seen for example in Figs. 7–9). By contrast, Fig. 11 illustrates the ability of the Wolff-Gibbs segmentation algorithm to avoid over-segmentation: the weight of the chart is at a markedly higher value of $1 - f$ than that in Fig. 10.

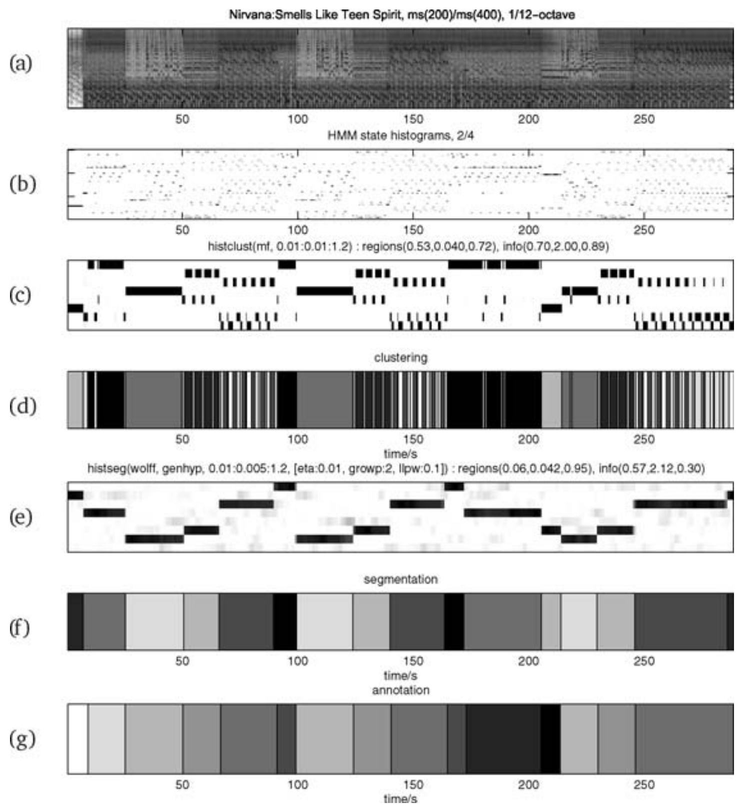

Fig. 9 A segmentation of the same song as in Fig. 8, but with differently-sized windows for the audio and HMM state sequence framing steps

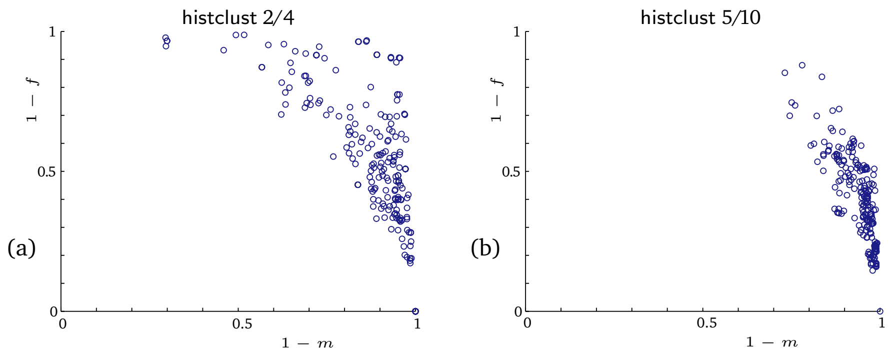

Fig. 10 Values of $1 - f$, corresponding loosely to precision, plotted against values of $1 - m$, analogous to recall, for the central clustering algorithm with hopsize/framesize 2/4 in (a), 5/10 in (b)

In Fig. 12 we plot values of f against the number of segments requested from each algorithm. Note that in each case an increase in the number of segments results in a greater f, and also that the Wolff algorithm performs very consistently by this measure over our test data.

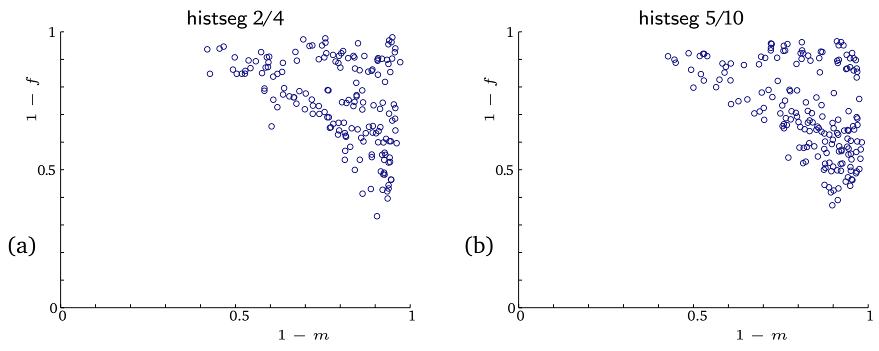

Fig. 11 Values of $1 - f$, corresponding loosely to precision, plotted against values of $1 - m$, analogous to recall, for the Wolff-Gibbs segmentation algorithm with hopsize/framesize 2/4 in (a), 5/10 in (b)

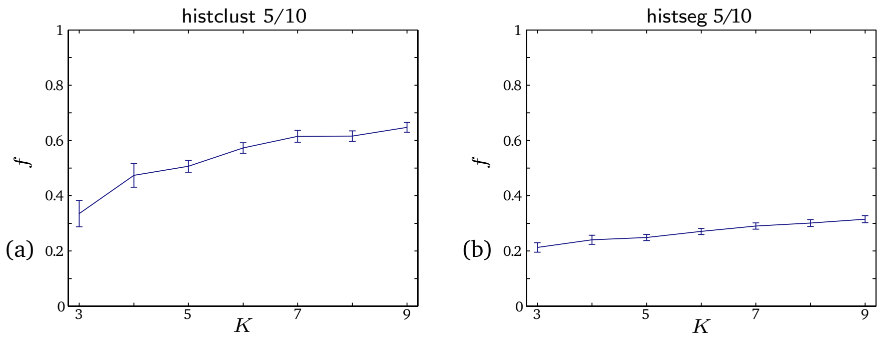

Fig. 12 Values of f plotted against the number of segments requested of each algorithm: (a) central clustering; (b) Wolff segmentation

We can use the mutual information measures described in Section 7.2.2 to display another view of the effect of increasing the number of class labels. In Fig. 13 we plot mutual information $I(\vec{c}_0, \vec{c}_1)$ and spurious information $H(\vec{c}_1 \mid \vec{c}_0)$ for one song in our data set, both for the central clustering and for the Wolff-Gibbs segmenter. The left panel shows that, while the Wolff-Gibbs algorithm performs slightly better, it is not by a very large margin (and indeed both algorithms fall quite a way short of the theoretical maximum); the right panel, however, illustrates the advantage of the Wolff-Gibbs algorithm: very little spurious information is added to a segmentation when increasing the number of segment labels, in contrast to the behaviour of the central clustering algorithm.

8 Discussion and conclusions

It is clear from the results presented that our modified Wolff algorithm incorporating an explicit prior on segment durations can be used to solve the problem of fragmentation in audio segmentation. By choosing a suitably broad prior distribution, we are able

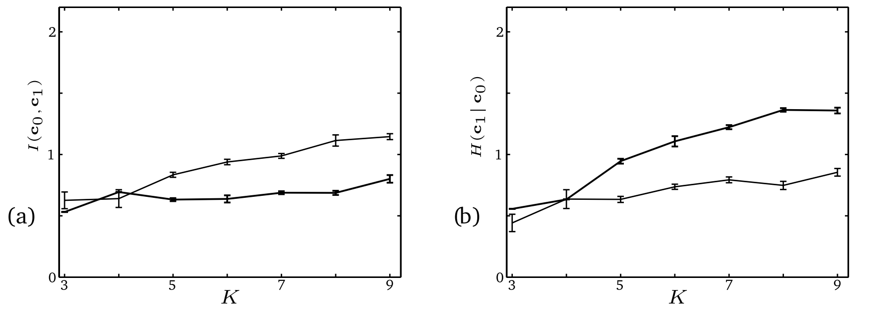

Fig. 13 Panel (a) shows the change of mutual information between segmentation and annotation $I(\vec{c}_0, \vec{c}_1)$ as a function of the number of clusters for one particular song in the corpus (whose ground truth annotation has 2.06 bits of entropy): the curve for the Wolff algorithm (thin line) is above that for the central clustering (thick line). The right panel shows the change of spurious information $H(\vec{c}_1 \mid \vec{c}_0)$ in the segmentation; here the central clustering curve is higher

to generate segmentations with a realistically wide range of segment lengths while avoiding the generation of a succession of short segments modelling rapidly varying features of the audio.

The crucial difference between the clustering approach and the duration prior approach is that the clustering approach operates in a feature space with the time dimension eliminated, gathering together audio frames with similar textural characteristics regardless of where they appear in the signal, and characterising each segment class as an average over a potentially disparate group of frames. The duration modelling approach forces the segment classes to be defined by averages over *temporally contiguous* frames. This can result in classes that, when projected into the timeless feature space, overlap significantly, yet retain their identity by virtue of temporal coherence.

This effect was clearly visible in some of the segmentations we obtained, for example, the song ‘Thank U’ by Alanis Morissette, (not shown here due to lack of space) in which much of the song consists of an alternating harmonic structure with some of the same chords being used in different sections. The histogram clustering analysis tended to result in a segmentation at the level of these harmonic variations (which

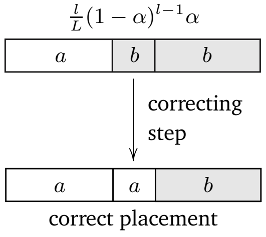

Fig. 14 Domain proposal for small boundary adjustment: the current segmentation *abb* has a slightly misplaced boundary, which can only corrected by selecting the short central domain with probability $\frac{l}{L}(1-\alpha)^{l-1}\alpha$. When α is small towards the end of the annealing schedule, the probability of selecting this domain decreases markedly, especially so if the length of the domain l is small, making it unlikely that the boundary correcting step will be taken

are well captured in a $\frac{1}{12}$th-octave constant-Q spectrogram), while the duration modelling segmenter was able to detect the different patterns of use of the same harmonic building blocks on a longer timescale.

One of the motivations in switching to explicit duration modelling rather than relying on large histogram windows—which is also effective in reducing fragmentation, as demonstrated, for example, by Abdallah et al. (2005)—was to improve the temporal resolution of the segmentation. However, we have found in our experiments that the Wolff domain proposal algorithm introduces a new source of inaccuracy in boundary placement. The problem can be understood with reference to Fig. 14: it is that the Markov chain steps necessary to correct minor errors in boundary placement become increasingly unlikey as the temperature of the simulation drops. We suggest ways of overcoming this problem in the next section.

8.1 Future work

8.1.1 Developments of the model

A relatively minor refinement to the model would be to introduce a richer parameterisation to be subjected to optimisation. One option would be to allow each segment class to adopt its own duration distribution, or at least its own characteristic timescale. Another would be to model the transition probabilities between different segment classes using a first-order Markov model. Both of these measures would help distinguish segment types that differ not in their textural aspects but in their patterns of duration and succession. However, both cases are likely to require an explicitly Bayesian (as opposed to maximum *a posteriori*) approach to learning in order to combat overfitting, since each song only has a rather limited number of segments and transitions from which to fit the model parameters.

Although we have begun to model temporal coherence of large-scale segments as in Section 5, we have not yet modelled the internal dynamics of such segments, merely their overall distribution over HMM states, which we treat as stationary. This prevents our current model from detecting consecutive repeats of the same segment class; two consecutive choruses will be detected as one long segment, not two repeats of the same type of segment. To address this, it would be necessary to model the typical evolution of each segment class. This could potentially be achieved by replacing each class-specific histogram model with a non-ergodic Markov model, such that each sub-model has a final state after which a new segment must be initiated.

Both of the above modifications would take us in the direction of more uniformly structured hierarchical HMMs, using the same mechanisms to model structures at different timescales.This would pave the way, in a unified and principled fashion, for the introduction of more levels in the hierarchy.

8.1.2 Developments of the MCMC algorithm

The performance of a MCMC-based system such as ours is heavily dependent on the proposal distributions used at each step of the Markov chain. As noted above, the Wolff-Gibbs algorithm has difficulty in correcting slightly misplaced boundaries towards the end of the annealing schedule because of the improbability of selecting the short domains that would be required to move the boundary (shown in Fig. 14). This problem could potentially be addressed in one or more of the following ways.

Firstly, a preliminary analysis of the audio signal using an onset detection system (e.g. Bello et al., 2004) or beat tracker could supply a map of likely boundary times which could be used to modulate the domain proposal distribution, biassing it towards domains that start and end at an onset or beat.

Even with these modifications, the MCMC approach is likely to remain much more computationally expensive than the deterministically-annealled system used on the histogram clustering model. Hence, we aim to investigate whether or not a similar mean field or variational EM approach might be taken with the duration-aware segmentation model, perhaps by reparameterising the segment configuration in terms of boundary times rather than as a regularly sampled sequence of frame classifications.

Appendix A: Notational conventions

In this paper, we adopt the following notation for type specifications:

$X : A$	Means that X is of type A, where A is some type specifier.
$D \rightarrow R$	The type of functions with domain D and range R.
$A^{N\times M}$	The type of $N \times M$ arrays with elements of type A.
A^*	The type of sequences of any length with elements of type A.
$\mathcal{D}(x)$	The domain of the function or array x.
$\mathcal{R}(x)$	The range of the function or array x.
$N..M$	The set or type of integers between N and M inclusive.
$\mathbb{R}_0^+$	The set or type of positive semi-definite reals $\{x \in \mathbb{R} \mid x \geq 0\}$.
$\mathsf{interval}(A)$	The type of intervals on the linearly ordered domain A.

In addition we define some notation for random variables. A probability space is a triple $(\Omega, \mathcal{F}, P)$, where Ω is a set of elementary events, $\mathcal{F}$ is a σ-field over Ω, and $P : \mathcal{F} \rightarrow \mathbb{R}_0^+$ is a measure such that $P(\Omega) = 1$. In these terms we can define two new types:

$\mathsf{rv}(A)$	A random variable with range A. If $X : \mathsf{rv}(\mathcal{X})$, then $X : \Omega \rightarrow \mathcal{X}$.
$\mathsf{pdf}(A)$	The type of probability density functions over a measurable space A, e.g., if $p : \mathsf{pdf}(\mathcal{X})$, then $p(x) \geq 0 \; \forall \, x \in \mathcal{X}$ and $\int_{\mathcal{X}} p(x) \mathrm{d}x = 1$.

Appendix B: Segment length prior

A prior given by the energy function (11), repeated here for convenience,

$$\varepsilon_{\mathcal{H}}(x, \nu, \gamma) = \frac{1}{|\nu|} x^{-\nu} + (\gamma + 1) \log x, \tag{26}$$

has probability distribution

$$p_{\mathcal{H}}(x, \nu, \gamma)\mathrm{d}x = \frac{1}{Z} e^{-\varepsilon_{\mathcal{H}}(x,\nu,\gamma)}\mathrm{d}x$$
$$= \frac{1}{Z} x^{-(\gamma+1)} e^{-\frac{x-\nu}{|\nu|}} \mathrm{d}x, \tag{27}$$

where Z is a normalizing constant.

The mode of this probability distribution is at the mininum of $\varepsilon_{\mathcal{H}}$ (since the exponential function is monotonic), which is given by

$$-\operatorname{sgn}(\nu) + \frac{\gamma + 1}{x} = 0. \tag{28}$$

This has no positive solutions for $\nu \leq 0$; for $\nu > 0$ the mode is at $x = \gamma + 1$. As for the constant of normalization,

$$Z = \int_0^\infty x^{-(\gamma+1)} e^{-\frac{x-\nu}{|\nu|}} \mathrm{d}x. \tag{29}$$

For $\nu = 0$, the distribution is not well-defined. Otherwise, we change variables to $u = \frac{x-\nu}{|\nu|}$, giving

$$Z = \int_0^\infty (|\ \nu\ |u)^{\frac{\gamma+1}{\nu}} e^{-u} (|\ \nu\ |u)^{-\frac{\nu+1}{\nu}} \mathrm{d}u$$
$$= \int_0^\infty (|\ \nu\ |u)^{\frac{\gamma-\nu}{\nu}} e^{-u} \mathrm{d}u$$
$$= |\ \nu\ |^{\frac{\gamma-\nu}{\nu}} \Gamma(\frac{\gamma}{\nu}). \tag{30}$$

Thus, the prior distribution (11) is improper for $\frac{\gamma}{\nu}$ being a nonpositive integer. However, for positive ν and zero γ, as used in this paper, the restricted prior obtained by cutting off the prior (11) at the length of the song being segmented, as discussed in Section 5.3, is a proper normalizeable prior.

Acknowledgments We thank the reviewers for their thoughtful and helpful comments. This research was supported by EPSRC grants GR/S84750/01 (Hierarchical Segmentation and Semantic Markup of Musical Signals) and GR/82213/01 (Information Dynamics of Music).

References

Abdallah, S., Noland, K., Sandler, M., Casey, M., & Rhodes, C. (2005). Theory and evaluation of a Bayesian music structure extractor. In J.D. Reiss & G.A. Wiggins (Eds), *Proceedings of the sixth international conference on music information retrieval*, (pp. 420–425).

Allen, J. (1984). Towards a general theory of action and time. *Artificial Intelligence*, *23*, 123–154.

Aucouturier, J.-J., Pachet, F., & Sandler, M. (2005). The way it sounds: Timbre models for analysis and retrieval of polyphonic music signals. *IEEE Transactions of Multimedia*.

Barbu, A. & Zhu, S.-C. (2004). Cluster sampling and its applications in image processing. Technical Report 409, Department of Statistics, UCLA.

Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., & Sandler, M. (2004). A tutorial on onset detection in music signals. *IEEE Transactions in Speech and Audio Processing*, *13*(5), 1035–1047.

Brown, J. C. (1991). Calculation of a constant Q spectral transform. *Journal of the Acoustic Society of America*, *89*(1), 425–434.

Dannenberg, R., & Hu, N. (2002). Discovering musical structure in audio recordings. In *Music and artifical intelligence: second international conference*. Edinburgh.

Downie, S., & Nelson, M. (2000). Evaluation of a simple and effective music information retrieval method. In *Proceedings of the ACM SIGIR* (pp. 73–80).

Eckmann, J.-P., Kamphorst, S. O., & Ruelle, D. (1987). Recurrence plots of dynamical systems. *Europhysics Letters*, *5*, 973–977.

Foote, J. (1999). Visualizing music and audio using self-similarity. In *ACM Multimedia*, vol. 1, pp. 77–80.

Galton, A. (Ed) (1987). *Temporal logics and their applications*. Academic Press, London.

Goto, M. (2003). A chorus-section detecting method for musical audio signals. In *Proc. ICASSP*, vol. V, pp. 437–440.

Hainsworth, S., & Macleod, M. (2003). Onset detection in musical audio signals. In *Proc. ICMC*.

Hofmann, T., & Buhmann, J. M. (1997). Pairwise data clustering by deterministic annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(1).

Huang, Q., & Dom, B. (1995). Quantitative methods of evaluating image segmentation. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP'95)*.

Logan, B. (2000). Mel frequency cepstral coefficients for music modeling. In *International Symposium on Music Information Retrieval*.

Logan, B., & Chu, S. (2000). Music summarization using key phrases. In *International Conference on Acoustics, Speech and Signal Processing*.

Lu, L., Wang, M., & Zhang, H. (2004). Repeating pattern discovery and structure analysis from acoustic music data. In *6th ACM SIGMM International Workshop on Multimedia Information Retrieval*.

Maddage, N., Changsheng, X., Kankanhalli, M., & Shao, X. (2004). Content-based music structure analysis with applications to music semantics understanding. In *6th ACM SIGMM International Workshop on Multimedia Information Retrieval*.

Merhav, N., & Lee, C.-H. (1993). On the asymptotic statistical behaviour of empirical cepstral coefficients. *IEEE Transactions on Signal Processing*, *41*(5), 1990–1993.

Orio, N., & Neve, G. (2005). Experiments on segmentation techniques for music documents indexing. In J. D. Reiss & G. A. Wiggins (Eds), *Proceedings of the sixth international conference on music information retrieval* (pp. 624–627).

Peeters, G., Burthe, A. L., & Rodet, X. (2002). Toward automatic music audio summary generation from signal analysis. In *International Symposium on Music Information Retrieval*.

Puzicha, J., Hofmann, T., & Buhmann, J. M. (1999). Histogram clustering for unsupervised image segmentation. *Proceedings of CVPR '99*.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selection applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286.

Robert, C. P., & Casella, G. (1999). *Monte carlo statistical methods*. Springer, New York.

Shoham, Y. (1988). *Reasoning about change: time and causation from the standpoint of artificial intelligence*. MIT Press, Cambridge, MA.

Swendsen, R. H., & Wang, J.-S. (1987). Non-universal critical dynamics in Monte-Carlo simulations. *Physical Review Letters*, *58*(2), 86–88.

Wakefield, G. H. (1999). Mathematical representation of joint time-chroma distributions. In *Advanced Signal Processing Algorithms, Architectures, and Implementations*, vol. 3807, IX, pp. 637–645. SPIE.

Wolff, U. (1989). Collective Monte Carlo updating for spin systems. *Physical Review Letters*, *62*(4), 361–364.