Inferring metrical structure in music using particle filters

Florian Krebs, Andre Holzapfel, Ali Taylan Cemgil and Gerhard Widmer

Abstract

In this work, we propose a new state-of-the-art particle filter (PF) system to infer the metrical structure of musical audio signals. The new inference method is designed to overcome the problem of PFs in multi-modal probability distributions, which arise due to tempo and phase ambiguities in musical rhythm representations. We compare the new method with a hidden Markov model (HMM) system and several other PF schemes in terms of performance, speed and scalability on several audio datasets. We demonstrate that using the proposed system the computational complexity can be reduced drastically in comparison to the HMM while maintaining the same order of beat tracking accuracy. Therefore, for the first time, the proposed system allows fast meter inference in a high-dimensional state space, spanned by the three components of tempo, type of rhythm, and position in a metric cycle.

Index Terms

Particle filters, Beat tracking, Downbeat tracking, Bayesian modeling, Approximate inference

I. Introduction

Automatic inference of metrical structure from musical audio has been an active research topic over the last few decades. Especially, estimating the perceptually most salient pulsation in musical meter, i.e., the beat, is one of the aspects that has attracted a significant amount of research work (see [1] for an

Manuscript received April 02, 2014; revised February 22, 2015.

The work was supported in part by the Austrian Science Fund (FWF) project Z159, by a Marie Curie Intra-European Fellowship (grant number 328379), by Boğaziçi University Research fund BAP 6882 12A01D5, and by the Turkish Research Council TUBITAK 113M492. Florian Krebs and Gerhard Widmer are with Johannes Kepler University, Linz. Andre Holzapfel and Ali Taylan Cemgil are with Boğaziçi University, Istanbul.

overview). While also interesting in its own right, the automatic determination of the metrical grid from an audio recording is a fundamental ingredient for other, high-level Music Information Retrieval (MIR) tasks, such as music genre recognition [2], chord estimation [3], and music transcription [4].

Over the last decade, probabilistic state-space models have become a popular framework to tackle the metrical inference problem (e.g., [5], [6], [7]). In these models it is attempted to infer a set of hidden states (such as beat times, tempo, meter) from a set of observed states (such as estimated note onset times and/or other audio features). Bayesian methods allow to easily represent the ambiguity that is inherent to musical meter and offer a consistent framework to combine multiple sources of information (e.g., preferred tempi, knowledge about note onset locations). Nevertheless, exact inference in these models is only feasible in a few simple cases, e.g., in discrete state spaces using hidden Markov models (HMMs) [8] or for linear Gaussian conditional distributions [9].

To overcome these limitations, approximative methods such as particle filters (PF) [10] have been proposed. Particle filters are a highly efficient method that can be applied to arbitrary, high-dimensional, non-linear, non-Gaussian state-spaces. Consequently, the use of particle methods makes it possible to use more complex models, which has also been exploited for the beat tracking task: [11], [12] and [13] model the beat times and tempo jointly, taking into account their mutual dependency. Furthermore, [14] introduced a rhythmic pattern state, which allows modeling various meters and rhythmic styles explicitly. In any case, the great flexibility of the PFs comes at a prize: Simple PF schemes are known to perform poorly with multi-modal probability distributions, which arise from ambiguity in the hidden state-sequence that generated the data [15]. Therefore, much work has been devoted to resolving this problem, and several extensions to the simple PF have been proposed [16], [15], [10]. However, although multi-modal distributions appear frequently in analysis of musical rhythm due to inherent tempo and phase ambiguities, dealing with this multi-modality with PFs in a robust way has never been addressed in the meter tracking literature.

In this work, we aim at setting a new state-of-the-art in PF based meter inference systems. We propose a new particle filter based inference method to overcome the problem of tracking a multi-modal distribution by combining the auxiliary (APF) [16] and the mixture particle filter (MPF) [15], and compare the resulting auxiliary MPF (AMPF) with the HMM system proposed in [17] and several other PF schemes in terms of performance, speed and scalability on several audio datasets. We demonstrate that using the AMPF the computational complexity can be reduced drastically compared to the HMM while maintaining the same order of beat tracking accuracy. Apart from beat tracking, our system is capable of determining downbeat times, time signature and type of rhythm of an audio piece using one integrated approach.

Thus, for the first time, the proposed AMPF permits fast inference in the high-dimensional state-space spanned by variables describing tempo, position within a metric cycle, and type of the metric cycle.

The structure of the underlying model presented in this paper can be easily adapted to the music style under investigation, and parameters of the model can be learned off-line from rhythmic patterns encountered in a representative music corpus. This will be described in more detail in Section III-D2. In most existing approaches, parameters and structure of the beat tracking systems are tailored by experts, with the goal to cope well with the demands of specific annotated evaluation datasets. This is problematic under at least two aspects. First, the manual adaption to a specific style is time-consuming and hence is not a viable solution for covering a wider variety of styles. Second, such systems are not suited for later adaption by users, and therefore incorporate the risk to include a bias [18] that systematically discriminates musical styles not considered during development. Therefore, our approach represents an important step towards flexible representations of musical concepts that can be adapted by incorporating available knowledge of musical style in a straight-forward way, as demanded in [19].

In the following sections, we will introduce basic notions of the metrical structure of music, and explain the structure of our dynamic Bayesian network by defining the hidden variables, the transition model and the observation model. In Section IV we describe different ways to perform inference in this model. First, the HMM (Section IV-A) represents an accurate inference approach with high computational cost. To reduce this cost, we describe several inference schemes based on PFs in Section IV-B. Section V describes datasets and evaluation metrics and the default settings for all system parameters. Experimental results are presented in Section VI, with a focus on the comparison of the HMM inference with the PF schemes. Section VII provides the reader with a detailed summary and discussion of the experimental results. The final Section VIII concludes the paper and outlines future directions of research motivated by our results.

II. Metrical structure of music

Meter as defined by [20] is organized pulsation functioning as a framework for rhythmic design. Especially in the context of Eurogenetic music, this pulsation is considered to be stratified into a hierarchy, with the period of pulsation increasing towards higher layers. The pulsation of the *beat* is situated in a layer in the middle of this hierarchy, and represents the rate that listeners choose to synchronize their body movements to the music. Beats are grouped into segments with a constant number of beats (*bars*), defined by the *time signature*. The first beat of each bar is denoted the *downbeat* (see Fig. 1 for an illustration). Listeners differ in their perception of meter, caused by individual or cultural factors [21]. For instance,

the perceived beat can be related to different metrical layers, which results in perceived tempi that are typically related by a factor of two. This variability must be considered both in the implementation of inference systems and in their evaluation.

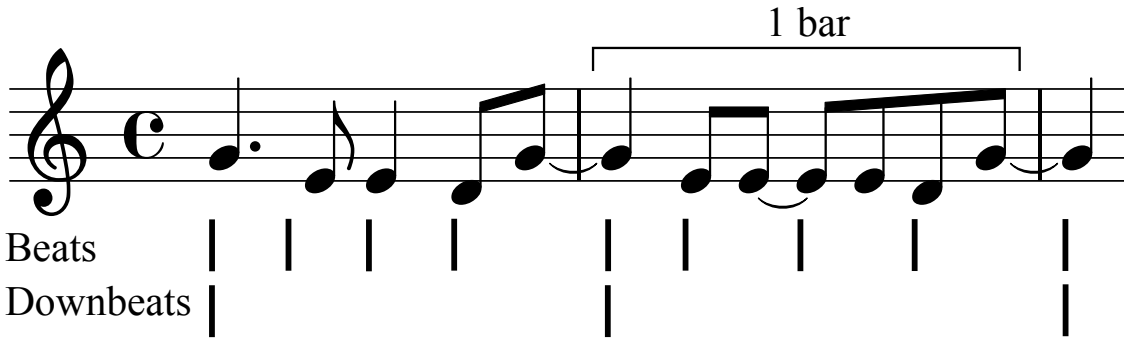

Fig. 1. Illustration of beats and downbeats in a musical score

Computational inference of meter can either be approached in an *on-line* or *off-line* fashion. On-line tracking requires inference at the same moment of observing the musical sound, without the possibility of looking into the future. On the other hand, off-line processing assumes that a recording of the whole piece is given, and for the meter inference at a specific moment in the recording the future can be taken into account. In this paper, we evaluate our system in off-line mode but the proposed methodology can be applied to on-line scenarios as well.

III. Model structure

In this section, we formulate the metrical structure analysis problem using a Bayesian model. We assume that a time series of *observed* data $\mathbf{y}_{1:K} = \{\mathbf{y}_1, ..., \mathbf{y}_K\}$ (the audio signal as a sequence of audio features) is generated by a set of unknown, *hidden* variables $\mathbf{x}_{1:K} = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$ (the parameters describing tempo and meter throughout the progression of a piece), where K is the length of an audio excerpt in analysis frames. In a *dynamic Bayesian network* (DBN) [22], the joint probability distribution of hidden and observed variables $P(\mathbf{y}_{1:K}, \mathbf{x}_{1:K})$ then factorizes as

$$P(\mathbf{y}_{1:K}, \mathbf{x}_{1:K}) = P(\mathbf{x}_1) \prod_{k=2}^{K} P(\mathbf{x}_k|\mathbf{x}_{k-1}) P(\mathbf{y}_k|\mathbf{x}_k), \quad (1)$$

where $P(\mathbf{x}_1)$ is the *initial state distribution*, $P(\mathbf{x}_k|\mathbf{x}_{k-1})$ is the *transition model*, and $P(\mathbf{y}_k|\mathbf{x}_k)$ is the *observation model*. We will describe these three terms in more detail in Sections III-B to III-D, after presenting the internal structure of the hidden variables in $\mathbf{x}$ in subsection III-A.

A. Hidden variables

The model described in this paper closely follows the bar pointer model proposed in [17]. In this model, the observation at each time step k is a short audio frame, and the hidden variables describe the state of a hypothetical bar pointer $\mathbf{x}_k = [\phi_k \ \dot{\phi}_k \ r_k]$ corresponding to the k-th audio frame; the variable ϕ_k is the current location in a bar, $\dot{\phi}_k$ is the instantaneous tempo (denoting the rate at which the bar pointer traverses a bar), and r_k is a rhythmic pattern indicator that can be used to differentiate between time signatures or between rhythmic styles of identical time signature. Below, we make these definitions more precise:

1) Bar position: We define the bar position $\phi \in [0, \theta_{max})$, where θ_{max} is the length of a bar related to the longest considered metric cycle in the data. For example, if the time signatures of the considered meters are $9/8$, $4/4$, and $3/4$, we set $\theta_{max} = 9/8$.

2) Tempo: We define tempo $\dot{\phi}_k \in [\dot{\phi}_{min}(r_k), \dot{\phi}_{max}(r_k)]$ in terms of beats per minute (bpm); The tempo limits are assumed to depend on the rhythmic pattern state r_k and are learned from data (e.g., for the rhythmic pattern variable r_k assigned to a Tango pattern we may find $\dot{\phi}_k \in [118bpm, 136bpm]$).

3) Rhythmic pattern: The rhythmic pattern variable $r_k \in [1...R]$ is an indicator, which selects one of the R underlying observation models. Each observation model is associated with a time signature $\theta(r)$ (e.g., $\theta(r) = 3/4$) and a specific rhythmic structure that is learned from data. Note that there can be several rhythmic patterns sharing the same time signature. An example of two such learned patterns is given in Fig. 4, along with a detailed description of the observation models in Section III-D.

The conditional independence relations between these variables are shown in Fig. 2. The hidden state sequence, inferred from an audio piece, can finally be translated into a sequence of *time signature(s)* $\theta(r_k)$, *downbeat times* (time frames that correspond to $\phi_k = 0$), and *beat times* (time frames that correspond to $\phi_k = i \cdot (denom(\theta(r_k)))^{-1}, i = 1, 2, ..., num(\theta(r_k))$, where $denom$ and num are the denominator and numerator, respectively).

B. Initial state distribution

Using the initial state distribution $P(\mathbf{x}_1)$, *a priori* knowledge regarding rhythmic aspects can be introduced into the system. The case that certain rhythmic patterns are encountered more frequently can be modeled, or certain tempo values can be preferred using a weighting function [5]. For the experiments in this paper, we have simply assumed uniformly distributed bar position, tempo, and rhythmic patterns states within the learned tempo ranges $[\dot{\phi}_{min}(r_k), \dot{\phi}_{max}(r_k)]$.

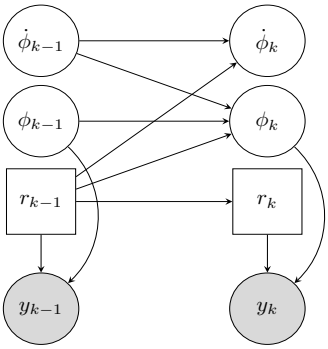

Fig. 2. Dynamic Bayesian network; circles denote continuous variables and rectangles discrete variables. The gray nodes are observed, and the white nodes represent the hidden variables.

C. Transition model

Due to the conditional independence relations shown in Fig. 2, the transition model factorizes as

$$\begin{aligned} P(\mathbf{x}_k|\mathbf{x}_{k-1}) = \quad & P(\phi_k|\phi_{k-1},\dot{\phi}_{k-1},r_{k-1})\times \\ & \times P(\dot{\phi}_k|\dot{\phi}_{k-1},r_{k-1}) \times P(r_k|r_{k-1}) \end{aligned} \tag{2}$$

where the three factors are defined by Equations 3-5:

$$P(\phi_k \mid \phi_{k-1},\dot{\phi}_{k-1},r_{k-1}) = \mathbb{1}_x, \tag{3}$$

where $\mathbb{1}_x$ is an indicator function that equals one if $\phi_k = (\phi_{k-1}+\dot{\phi}_{k-1}\cdot\Delta\cdot(60\cdot denom(\theta(r_{k-1})))^{-1})$ mod $\theta(r_{k-1})$, and zero otherwise, with $\Delta = 0.02s$ the audio frame length used in this paper. This means that the bar position at frame k is obtained by increasing the bar position of the previous frame by a term that depends on the tempo of the previous frame. The tempo transition from one frame to the next is assumed to follow a normal distribution and is given by

$$P(\dot{\phi}_k|\dot{\phi}_{k-1},r_{k-1}) \propto \mathcal{N}(\dot{\phi}_{k-1},\sigma^2_{\dot{\phi}}) \times \mathbb{1}_y, \tag{4}$$

where $\sigma_{\dot{\phi}}$ is the standard deviation of the tempo transition model and $\mathbb{1}_y$ is an indicator function that equals one if $\dot{\phi}_{min}(r_{k-1}) \leq \dot{\phi}_k \leq \dot{\phi}_{max}(r_{k-1})$, and zero otherwise.

$$P(r_k|r_{k-1}) = \mathbb{1}_z, \tag{5}$$

where $\mathbb{1}_z$ is an indicator function that equals one if $r_{k+1} = r_k$, and zero otherwise. This means that we assume a musical piece to have a characteristic rhythmic pattern that remains constant throughout the

song. This rather strict assumption can be relaxed by defining a rhythmic pattern transition probability that is nonzero at bar boundaries as in [17].

D. Observation model

As proposed in [23], we use an onset feature as observed variable $\mathbf{y}_k$, which we assume to be independent of the current tempo $\dot{\phi}_k$. Therefore, the observation model $P(\mathbf{y}_k|\mathbf{x}_k)$ reduces to $P(\mathbf{y}_k|\phi_k, r_k)$, which means that the probability of observing a certain feature value at a given timepoint depends only on the rhythmic style and the position in a bar. In order to obtain the parameters of $P(\mathbf{y}_k|\phi_k, r_k)$, a collection of beat- and downbeat annotated audio samples is needed (see Section V-C for details on the training set we use in this paper).

1) Observation features: As observation feature, we use a variant of the *LogFiltSpecFlux* onset feature, which performed well in recent comparisons of onset detection functions [24] and is summarized in Fig. 3. Assuming that the bass instruments play an important role in defining rhythmic patterns, we compute the sum over frequency bands separately in low frequencies (below 250 Hz) and high frequencies (above 250 Hz). Finally, we subtract the moving average computed over a window of one second and normalize the features of each excerpt to zero mean and unit variance, again separately for the low and high frequencies. The resulting onset feature $\mathbf{y}_k$ has therefore two dimensions.

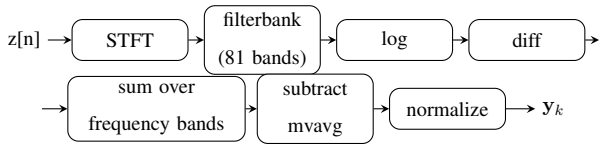

Fig. 3. Computing the onset feature $\mathbf{y}_k$ from the audio signal $z[n]$

2) Likelihood function: The parameters of the observation model are obtained in an off-line (supervised) rhythm pattern learning process. In order to get a modest (and computationally feasible) number of probability distributions to represent $P(\mathbf{y}_k|\phi_k, r_k)$, we discretize the bar position Φ into 64th note cells. In order to learn the parameters of the likelihood function, a set of training pieces must be available that is annotated at the beat and bar layers of the metrical hierarchy. Using these annotations, we can assign each feature vector $\mathbf{y}_k$ to the corresponding rhythmic pattern and bar position within the 64th note grid. Then, for each bar position within this 64th grid and each rhythmic pattern, we compute the maximum likelihood estimates of the parameters in a Gaussian mixture model (GMM). As suggested in [23], we

use $I = 2$ mixture components in this work. Hence, the observation probability is modeled by

$$P(\mathbf{y}|\phi, r) = \sum_{i=1}^{I} w_{\phi,r,i} \cdot \mathcal{N}(\mathbf{y}; \mu_{\phi,r,i}, \Sigma_{\phi,r,i}), \tag{6}$$

where $\boldsymbol{\mu}_{\phi,r,i}$ is the (2-dimensional) mean vector, $\boldsymbol{\Sigma}_{\phi,r,i}$ is the (2×2) covariance matrix, and $w_{\phi,r,i}$ is the mixture weight of component i of the GMM. Since, in learning the likelihood function $P(\mathbf{y}|\phi, r)$, a GMM is fitted to the audio features for every rhythmic pattern r and each 64th note bar position cell, the resulting GMMs can be interpreted directly as representations of the rhythmic patterns in the domain of the observation variables. Fig. 4 shows the mean values of the features per frequency band and bar position for the GMMs corresponding to the rhythmic patterns of a Waltz (3/4) and a Tango dance (4/4), respectively. For illustration purposes we only display the mean value of the *features* instead of the mean values per mixture component.

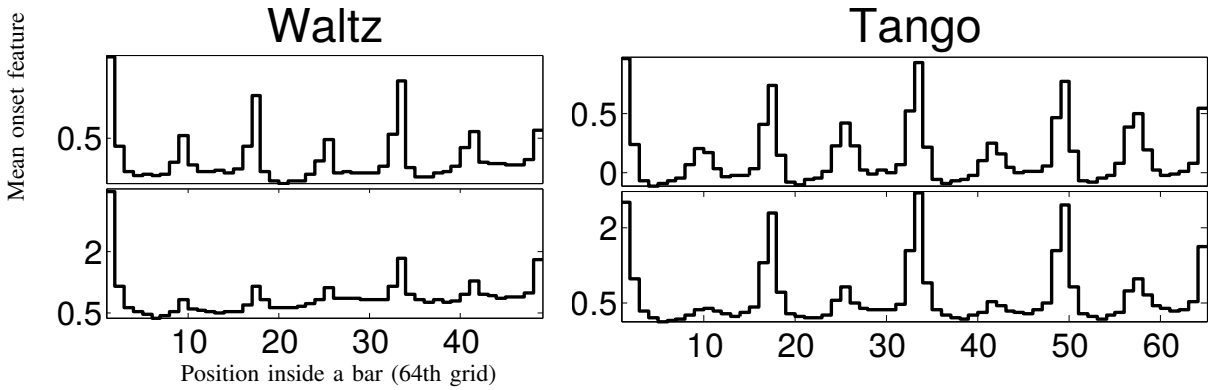

Fig. 4. Illustration of two learned rhythmic patterns. Two frequency bands are shown (Low/High from bottom to top).

IV. Inference methods

Our goal is to find the hidden state sequence that maximizes the (posterior) probability of the hidden states given the observations $P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$. If we discretize the continuous tempo and bar pointer variables (see Section III-A), we can in principle perform an exact inference using an HMM (Section IV-A).

However, in order to avoid the high computational complexity of the HMM inference illustrated in Section IV-A, we describe approaches for inference using PFs in Section IV-B. We begin with PF approaches widely discussed in literature, and then present novel approaches capable of tracking the metrical structure in the highly multi-modal posterior distributions typical for musical rhythm.

A. Hidden Markov Model (HMM)

Inference in the model discussed in Section III can be performed using an HMM by dividing the state space into discrete cells and using Viterbi decoding [8] to obtain the *maximum a posteriori* (MAP) sequence of states. In Fig. 5 we show a realization of a bar position/tempo trajectory and a possible discretization.

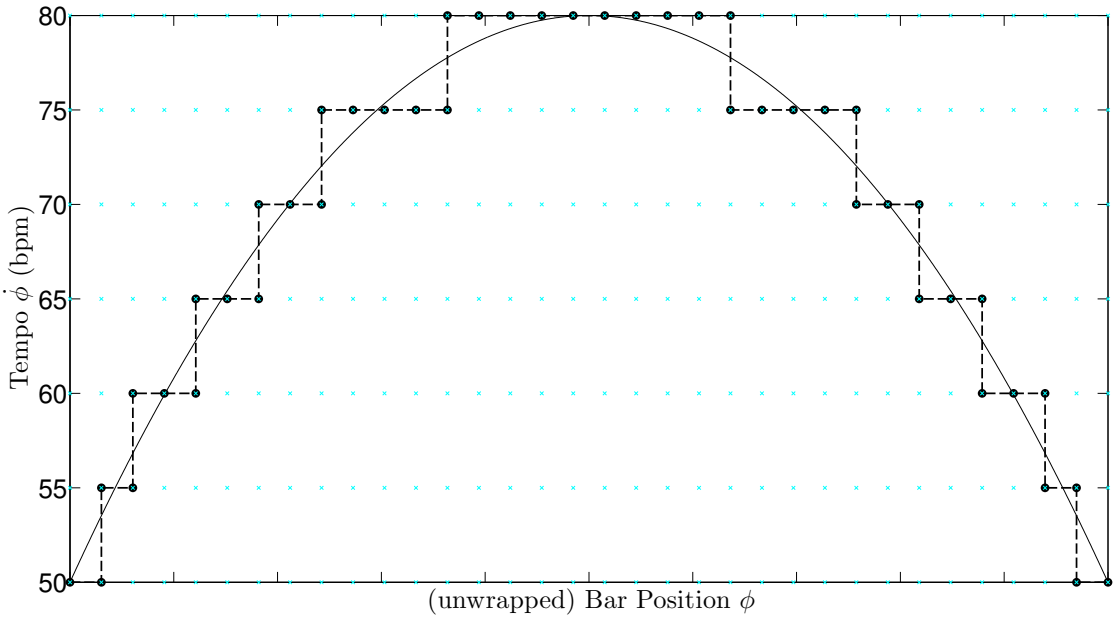

Fig. 5. Illustration of discretization of the tempo and bar pointer variables. The continuous line depicts a tempo trajectory that might be encountered in the expressive timing throughout a musical phrase, and the the dashed line demarks the trajectory through the discretized states.

In this work, we use the discretization proposed in [17], which we further explain in this section. By replacing the continuous variables ϕ and $\dot{\phi}$ by their discretized counterparts $m \in \{1, ..., M\}$ and $n \in \{1, ..., N\}$ respectively, Equations 2, 3 and 5 remain valid. We only define a new tempo transition probability as:

If $n_{min}(r_k) \leq n_k \leq n_{max}(r_k)$,

$$P(n_k|n_{k-1}) = \begin{cases} 1 - p_n, & n_k = n_{k-1}; \\ \frac{p_n}{2}, & n_k = n_{k-1} + 1; \\ \frac{p_n}{2}, & n_k = n_{k-1} - 1, \end{cases} \tag{7}$$

otherwise $P(n_k|n_{k-1}) = 0$.

Here, p_n is the probability of a tempo change and $n_{min}(r_k)$ and $n_{max}(r_k)$ are the discrete tempo limits that correspond to $\dot{\phi}_{min}$ and $\dot{\phi}_{max}$.

The HMM is most accurate when the discretization grid is dense, but this can become computationally prohibitive. For instance, discretizing the bar position ϕ of a 4/4 bar into $M = 1200$ points (300 per quarter note) and the tempo $\dot{\phi}$ into 23 points, results in a state-space of $S = M \cdot N = 27\,968$. Adding more rhythmic pattern states increases the dimensionality to $M \cdot N \cdot R$, which quickly surpasses the computational and memory limits of a current personal computer. This problem can be overcome by applying approximate inference schemes instead of the exact HMM inference, and we will present such schemes in the following parts of this section.

B. Particle filter (PF)

Even though the exact computation of the posterior $P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$ in the continuous parameter space is intractable, it can nevertheless be evaluated point-wise. This fact is exploited in the PF where the posterior is approximated by a weighted sum of points (i.e., particles) in the state space as

$$P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K}) \approx \sum_{i=1}^{N_s} w_K^{(i)} \delta(\mathbf{x}_{1:K} - \mathbf{x}_{1:K}^{(i)}). \tag{8}$$

Here, $\{\mathbf{x}_{1:K}^{(i)}, i = 1, ..., N_s\}$ is a set of points with associated weights $\{w_K^{(i)}, i = 1, ..., N_s\}$ and $\mathbf{x}_{1:K}$ is the set of all states until time frame K, while $\delta(x)$ denotes the Dirac Delta function

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{if } x \neq 0. \end{cases} \tag{9}$$

In order to approximate $P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$ we need a strategy to draw samples $\mathbf{x}_k^{(i)}$ and compute appropriate weights $w_k^{(i)}$ recursively for each time k. A simple algorithm to do that for sequential data is *sequential importance sampling* (SIS) [10]. In SIS, we sample from a *proposal* distribution $Q(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$ which should be as similar as possible to the true (*target*) distribution $P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$. To correct for the fact that we sampled from the proposal instead of the target distribution, we assign an *importance weight* $w_K^{(i)}$ to each particle, which is computed by

$$w_K^{(i)} = \frac{P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})}{Q(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})}. \tag{10}$$

If a suitable proposal density is chosen, these weights can be computed recursively by

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{P(\mathbf{y}_k|\mathbf{x}_k^{(i)}) P(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{Q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k)}. \tag{11}$$

In this work, we chose to sample from the transition probability $Q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k) = P(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})$, which reduces Eq. 11 to

$$w_k^{(i)} \propto w_{k-1}^{(i)} P(\mathbf{y}_k|\mathbf{x}_k^{(i)}). \tag{12}$$

Then, the SIS algorithm derives samples and weights for time k by first drawing from the proposal, in our case $P(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})$, and then assigning weights according to (12).

Once the particle trajectories $\{\mathbf{x}_{1:K}\}$ have been determined, we select the particle trajectory $\mathbf{x}_{1:K}^{(i)}$ with the highest weight $w_K^{(i)}$ as the MAP state sequence. We have not attempted to improve the obtained state sequence by particle Viterbi decoding [25] or particle smoothing [10] and leave this for future work.

Several extensions to the SIS filter have been proposed over the years (see [10] for a good overview). In the following, we will describe those approaches that we will evaluate for their applicability to metrical inference.

1) The sequential importance sampling/resampling (SISR) filter: The most challenging problem in particle filtering is to cope with the so called *degeneracy problem* [10]: After some time, most of the particles have a weight close to zero, and thus represent very unlikely regions of the state space. This is in contrast to the ideal case with a perfect match between the proposal and target distribution, where the weights are uniformly distributed and thus have a low variance. In order to reduce the variance of the particles, it has been recommended to use *resampling* or *rejuvenation* steps, in order to replace particles with low weights by particles with a higher weight. This is usually done by selecting particles with a probability that is proportional to their weights. Several schemes have been proposed in the literature, for a comparison see [26]. The most common resampling method is *systematic resampling* [27] and has also been used in this paper. As recommended in [10], we do not resample at each iteration (*bootstrap filter*), but only perform resampling when the effective sample size

$$N_{ESS} = (\sum_{i=1}^{N_s} (w_k^{(i)})^2)^{-1} \tag{13}$$

is below a threshold of $\rho \cdot N_s$. The value of ρ will be determined together with other system parameters in Section V-C.

The essential difference between SIS and SISR is therefore the resampling. The SISR algorithm is outlined in Algorithm 1. Note that we sample the tempo state $\dot{\phi}$ at the end of each iteration (after the resampling step). This is motivated by the general principle that any operation that does not influence the weights should take place after the resampling step in order to increase the diversity of the particles [10].

Algorithm 1 Outline of the sequential importance sampling/resampling (SISR) filter, $\xi_k^{(i)}$ denotes $(\phi_k^{(i)}, r_k^{(i)})$

for $i = 1$ to N_s **do**
  Sample $(\phi_0^{(i)}, \dot{\phi}_0^{(i)}, r_0^{(i)}) \sim P(\phi_0)P(\dot{\phi}_0)P(r_0)$
  Set $w_0^{(i)} = 1/N_s$
end for
for $k = 1$ to K **do**
  for $i = 1$ to N_s **do** ▷ Proposal and weight computation
    Sample $\xi_k^{(i)} \sim P(\xi_k^{(i)}|\xi_{k-1}^{(i)})$
    $\tilde{w}_k^{(i)} = w_{k-1}^{(i)} \times P(\mathbf{y}_k|\xi_k^{(i)})$
  end for
  for $i = 1$ to N_s **do** ▷ Normalize weights
    $w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{i=1}^{N_s} \tilde{w}_k^{(i)}}$
  end for
  Compute the effective sample size N_{ESS} (13)
  if $N_{ESS} \leq \rho \cdot N_s$ **then**
    for $i = 1$ to N_s **do**
      Resample $\{\xi_k^{(i)}, w_k^{(i)}\}$ to obtain $\{\xi_k'^{(i)}, 1/N_s\}$
    end for
    for $i = 1$ to N_s **do**
      $\xi_k^{(i)} = \xi_k'^{(i)}$
    end for
  end if
  Sample $\dot{\phi}_k^{(i)} \sim P(\dot{\phi}_k|\dot{\phi}_{k-1}^{(i)}, r_k^{(i)})$
end for

2) The Auxiliary Particle Filter (APF): Although the introduction of resampling steps reduces the variance of the importance weights, the degeneracy problem is not yet solved. Resampling tends to lead to an extreme concentration of the particles at one particular mode of the posterior distribution, whereas the remaining distribution remains uncovered. This is particularly critical if the probability distribution has multiple modes as it is the case in our application.

One way to alleviate this problem is to compress the weights $\mathbf{w}_k = \{w_k^{(j)}, j = 1, ..., N_s\}$ by a monotonically increasing function g[1] before resampling. This increases the weights of particles at low probability regions and therefore makes it more probable that these particles survive the resampling. After resampling, the weights have to be uncompressed again in order to yield a valid probability distribution. This can be formulated in the terminology of the *auxiliary particle filter* (APF) [28], which can be summarized as follows:

- Compute the compressed weights by applying $g(\cdot)$, which causes a decrease of variance in the weights.
- Resample particles according to the compressed weights.
- Set each weight to the quotient of the uncompressed and the compressed weight.

The resampling procedure of the APF is sketched in Algorithm 2.

As an example, imagine a distribution of two particles $\{p_1, p_2\}$ with corresponding weights $\mathbf{w} = \{0.01, 0.99\}$. If we drew two samples from this distribution, it would be probable that p_1 vanishes from the particle set at the resampling step. However, if we modify the weights by the function $f(x) = x^{\frac{1}{4}}$ yielding $\hat{\mathbf{w}} \approx \{0.32, 1.00\}$, drawing p_1 becomes much more probable. Let us assume, we draw four samples from this (unnormalized) distribution $\hat{\mathbf{w}}$ and obtain the set $\{p_2, p_1, p_2, p_2\}$). In order to still represent the same distribution as before the sampling, we have to set the weight of each resampled particle to $x/f(x)$, which yields $\mathbf{w} \approx \{0.99, 0.03, 0.99, 0.99\}$.

3) Mixture particle filter (MPF): As mentioned before, one major problem with applying particle filtering to the musical meter tracking problem is that the posterior distribution $P(\mathbf{x}_k|\mathbf{y}_{1:k})$ is highly multi-modal. A system that is able to cope with metrical ambiguities should maintain this multi-modality and track several hypotheses over a longer time.

In [15] a system was proposed that tracks multiple football players in a video sequence using a *mixture PF*. Each particle is assigned to a cluster based on its location within the state space. Whenever it comes to resampling, particles interact only with particles of the same cluster (resampling of one cluster is performed independently of all other clusters). In this way, all modes that are covered by a cluster can be tracked successfully. In the following we describe an adaption of this method to the problem of finding the metrical structure in music.

At the beginning, we cluster the particles into C_0 clusters[2] by a run of the k-means clustering algorithm

[1]In this work, we restrict $g(w)$ to functions of the form w^β where $0 \leq \beta < 1$.

[2]In this paper, we start with 16 clusters per rhythmic pattern ($C_0 = 16 \cdot R$).

Algorithm 2 Outline of the resampling procedure of the auxiliary particle filter (APF), $\xi_k^{(i)}$ denotes $(\phi_k^{(i)}, r_k^{(i)})$

Compute the effective sample size N_{ESS} (13)
if $N_{ESS} \leq \rho \cdot N_s$ **then**
  for $i = 1$ to N_s **do**
    $\hat{w}_k^{(i)} = w_k^{(i)} g(\xi_k^{(i)})$
  end for
  for $i = 1$ to N_s **do**
    Resample $\{\xi_k^{(i)}, \hat{w}_k^{(i)}\}$ to obtain $\{\xi_k'^{(i)}, w_k^{(i)}/\hat{w}_k^{(i)}\}$
  end for
  for $i = 1$ to N_s **do**
    $\xi_k^{(i)} = \xi_k'^{(i)}$
  end for
end if

using a distance measure which takes into account the cyclic nature of the bar position ϕ. This means that a point at the beginning of a bar ($\phi \approx 0$) should be close to a point at the bar ending ($\phi \approx \theta$). Therefore, we represent the bar position as a complex phasor on the unit circle and compute the corresponding angle by

$$\alpha_k(\phi_k, r_k) = \frac{2\pi\phi_k}{\theta(r_k)}. \tag{14}$$

This angle can further be expressed by the periodic functions $\cos(\alpha)$ and $\sin(\alpha)$ using Euler's formula

$$e^{j\alpha} = \cos(\alpha) + j\sin(\alpha). \tag{15}$$

Using this transformation we define the distance measure for the k-means clustering algorithm as

$$\begin{aligned} d(i,j) =& \lambda_\phi \cdot [(\cos(\alpha^{(i)}) - \cos(\alpha^{(j)}))^2 + (\sin(\alpha^{(i)}) - \sin(\alpha^{(j)}))^2] + \\ & \lambda_{\dot{\phi}} \cdot (\dot{\phi}^{(i)} - \dot{\phi}^{(j)})^2 + \lambda_r \cdot (r^{(i)} - r^{(j)})^2, \end{aligned}$$

where $[\alpha^{(i)}, \dot{\phi}^{(i)}, r^{(i)}]$ are the coordinates of the *i*th particle, and $\lambda_\phi, \lambda_{\dot{\phi}}, \lambda_r$ are coefficients that control the relative distance between the hidden variables. The assignment of particles to clusters is preserved until the next resampling step. Then, before resampling, the particles are reclustered in another run of k-means, using the old cluster assignments as initialization. Additionally, clusters are split, if the average distance from particle to cluster centroid is above a threshold τ_s and/or merged if the distance between

two centroids is below a threshold τ_m. If the number of clusters exceeds a constant τ_c[3], the clusters with the lowest total weight are removed, assigning the affected particles to other clusters. These three operations are important to control the number of clusters, which should ideally represent the number of modes of the posterior distribution. In order to have a balanced number of particles per cluster, we draw the same number of particles for each cluster in the resampling step.

In contrast to the SISR and APF approaches, it does not make sense to determine the timing of the resampling based on the effective sample size N_{ESS}. If we computed the N_{ESS} on the whole particle set, mixture components with low total weights would constantly lead to a low N_{ESS} and therefore to more frequent resampling steps. Therefore, we chose to perform the resampling step with a fixed interval d.

4) Auxiliary Mixture particle filter (AMPF): The AMPF combines the compression/decompression of the importance weights of the APF with the mixture tracking of the MPF.

V. Experimental Setup

A. Datasets

The performance evaluation of the PF algorithms and the HMM reference system requires annotated music recordings. We use three datasets for evaluation, which are frequently used in the MIR research community, and one for training:

The *SMC* dataset was presented in [29] and consists of 217 pieces that were considered to be difficult for automatic tempo tracking. Musical styles cover, e.g., French Chanson, classical orchestra music, and contemporary guitar music. For the *SMC* dataset, only beat annotations are available.

The largest dataset consists of 1360 songs and was compiled by Gouyon [30] combining collections from various sources. We will refer to this dataset as the *1360-song* dataset throughout the text. It contains a wide range of musical styles of mainly Eurogenetic music, covering choral works of classical music as well as rock or jazz. The dataset is only annotated at the beat-level.

The third dataset used for evaluation is the *Ballroom* dataset, introduced in [31] and annotated in [23]. It contains 698 excerpts of ballroom dance music, along with dance style, beat, and downbeat annotations, which enables us to evaluate at both metrical levels. We removed 13 replicated excerpts[4] to yield 685 unique ones. For some experiments, the dataset was split randomly into a test set (denoted *Ballroom_test*) of 204 songs and a training set (*Ballroom_train*) which consists of the remaining 481 songs, both having

[3]Here, we used a maximum number of 50 clusters per rhythmic pattern ($\tau_c = 50 \cdot R$).

[4]http://media.aau.dk/null_space_pursuits/2014/01/ballroom-dataset.html

roughly the same genre distribution. The audio quality of this dataset is quite low (RealAudio format with high compression).

The fourth dataset which was used for training only is a collection of 97 files from the MIREX 2006 beat tracking contest, from [32], and from [24]. We denote this dataset as the *MBB* dataset. It covers the genres pop, rock and electronic music and is beat and downbeat annotated.

Due to the lack of downbeat annotations for the *SMC* and *1360-song* datasets, downbeat detection is only evaluated on the *Ballroom* dataset.

B. Evaluation measures

We evaluate the HMM and PF inference schemes with respect to both *tracking accuracy* and *runtime*.

A variety of measures for beat and downbeat tracking performance is available (see [33] for a detailed overview). We chose four metrics that are characterized by a set of diverse properties and that are widely used in beat/downbeat tracking evaluation[5]. Furthermore, their choice enables for a direct comparison of the results on the *SMC* and the *1360-song* dataset with 16 reference systems [29].

1) F-measure: The F-measure is computed from correctly detected beats within a window of ± 70 ms by

$$\text{F-measure} = \frac{2pr}{p+r} \tag{16}$$

where p (*precision*) denotes the ratio between correctly detected beats and all detected beats, and r ($recall$) denotes the ratio between correctly detected beats and the total number of annotated beats. The range of this measure is from 0% to 100%.

2) AMLt (Allowed Metrical Level with no continuity required): In this method an estimated beat is counted as correct, if it lies within a small tolerance window around an annotated pulse, and the previous estimated beat lies within the tolerance window around the previous annotated pulse. The value of this measure is then the ratio between the number of correctly estimated beats divided by the number of annotated beats (as percentage between 0% and 100%). Beat sequences are also considered as correct if the beats occur on the off-beat, or are tapped at double or half the annotated tempo.

3) Information gain: This measure is computed by calculating the timing errors between an annotation and all beat estimations within a one-beat length window around the annotation. Then, a beat error histogram is created from the resulting timing error sequence. A numerical score is derived by measuring

[5]We used the MATLAB evaluation code available at http://code.soundsoftware.ac.uk/projects/beat-evaluation/ with standard settings to ensure reproducibility.

the K-L divergence between the observed error histogram and the uniform distribution. This method gives a measure of how much information the beats provide about the annotations. The range of values for the Information Gain is 0 bits to approximately 5.3 ($= \log_2(40)$) bits, for 40 histogram bins.

4) Downbeat F-measure: For measuring the downbeat tracking performance, we use the same F-measure as for beat tracking (using a ± 70 ms tolerance window).

As our focus in this paper lies on a proof-of-concept for PF-based methods, we present the mean values across all files of a dataset, without analyzing correlations between accuracies and certain musical styles. We postpone such a more musical interpretation and its implications to a future publication. As the result of a PF is a random variable itself (due to the stochastic nature of a PF), we have carried out each experiment in the next section ten times and give the mean and the standard deviation across these ten trials for all PF approaches.

The specified *runtimes* were measured using a MATLAB implementation of the systems on a PC with an Intel Core i5-2400 CPU with 3.1GHz. They include the computation of beats, downbeats and meter from the test dataset and exclude feature extraction and training which is the same for all models (e.g., for the Ballroom dataset feature extraction takes approximately 10 minutes and training the GMMs takes 30 seconds).

C. Determining system parameters

Both HMM and PF systems have a set of parameters that need to be determined. In the following we explain them in detail:

1) Observation model: For each test set, we learned the parameters of the observation model (see Section III-D) from a non-overlapping training set, as shown in Table I. This implies that for a given test set all PF and HMM methods use the same observation model.

TABLE I
TRAIN AND CORRESPONDING TEST DATASETS.

Train set	Test set
Ballroom+MBB	SMC
Ballroom+MBB	1360-song
Ballroom_train	Ballroom_test

2) Number of discrete states of the HMM: As explained in Section IV-A, the performance of the HMM depends on the density of the grid formed by the discretized variables. To visualize this dependency we illustrate the beat tracking accuracy and runtime of the HMM system for various state space sizes on the MBB training set in Fig. 6. In experiment 1, we will use two rhythmic patterns ($R = 2$, one for each of the two time signatures in the data), and in experiment 2, we will use eight rhythmic patterns ($R = 8$, one for each ballroom dance style). Considering both beat tracking accuracy and runtime, we chose to report results for three configurations of the HMM (depicted as HMM1-HMM3 in Table II). Note that the largest model (HMM3) with $R = 2$ has a number of discrete states equal to $M \times N \times R = 209\,152$ and is therefore situated in the middle range of values depicted in Fig. 6.

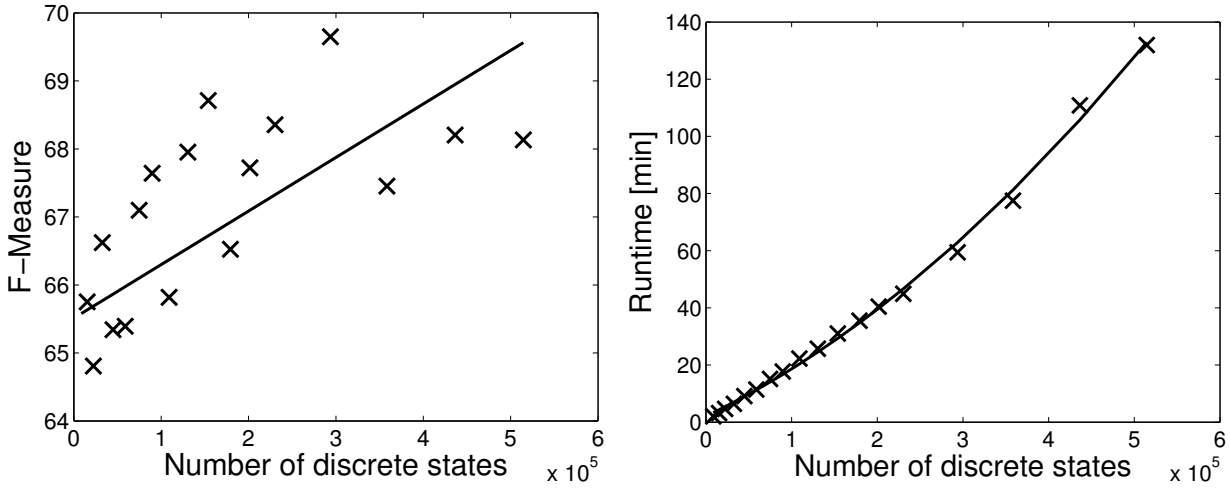

Fig. 6. F-measure and runtime vs. number of discrete states for the HMM on the MBB set (total 41.9 minutes of audio).

3) Number of particles of the PFs: In all PF variants we used $N_S = 2\,000$ particles in experiment 1, and $N_S = 8\,000$ in experiment 2. As can be seen in Fig. 7, increasing the number of particles above this value did not further improve the performance. Also note the differences in runtime between AMPF and HMM by comparing Fig. 7 with Fig. 6, with a clear advantage for the PF which will be further documented in the next section on the larger evaluation datasets.

Apart from the design choices above, we obtained parameter values by performing a simple grid search over the parameter space using the MBB dataset and selecting the overall best performing parameters according to the three beat and one downbeat tracking measures. The determined parameters are:

- Tempo transition probability p_n of the HMM
- Resampling threshhold ρ of the SISR and APF approaches

- Parameters of the MPF/AMPF $\lambda_\phi, \lambda_{\dot{\phi}}$, λ_r, τ_m, τ_s, d, β (see Section IV-B3)

For the PFs, we repeated each experiment ten times and averaged the results. The selected parameters are shown in Table II.

TABLE II
SELECTED PARAMETERS FOR THE EXPERIMENTS.

	ρ	d	$\sigma_{\dot{\phi}}$	λ_ϕ	$\lambda_{\dot{\phi}}$	λ_r	τ_m	τ_s	β
SISR	0.02	-	1.2	-	-	-	-	-	-
APF	0.1	-	1.2	-	-	-	-	-	1/5
MPF	-	30	1.2	1	1.4	1000	1	1.2	-
AMPF	-	30	1.2	1	1.4	1000	1	1.2	1/4
	M	N	p_n						
HMM1	640	12	0.02						
HMM2	1216	23	0.02						
HMM3	2432	43	0.02						

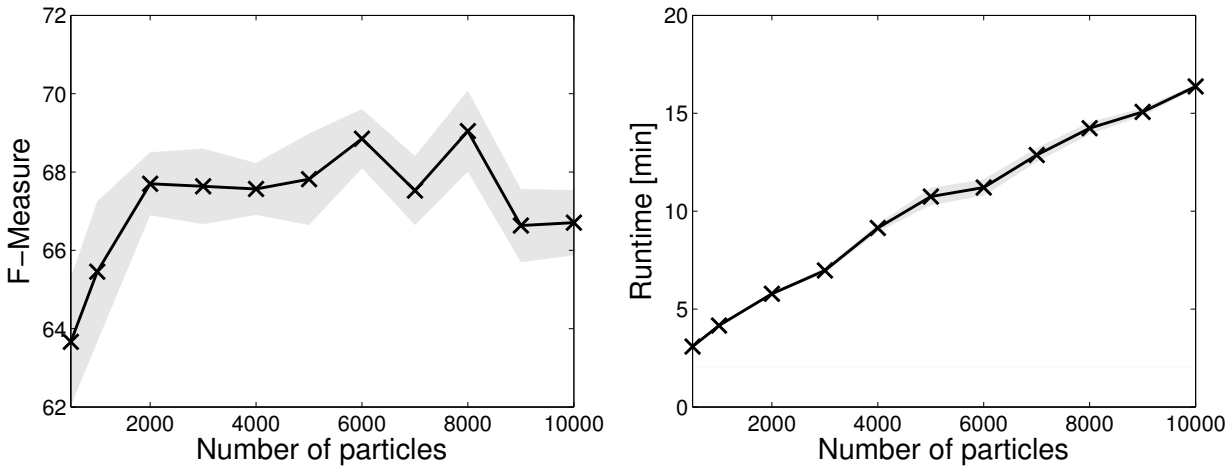

Fig. 7. F-measure and runtime vs. number of particles for the AMPF on the MBB dataset (total 41.9 minutes). The obtained standard deviation across ten runs is depicted by the shaded gray area.

VI. EXPERIMENTS

A. Experiment 1: PF against HMM

Experiment 1 compares the four PF inference schemes (SISR, APF, MPF, AMPF) with the HMM inference, in terms of their runtime complexity, beat (all datasets) and downbeat (only *Ballroom* dataset)

tracking accuracy. Tables III to V summarize the mean accuracies (and standard deviations for PF schemes) using all evaluation metrics on the three evaluation datasets.

TABLE III

BEAT TRACKING RESULTS AND (AVERAGE) RUNTIMES ON THE SMC DATASET (TOTAL 144.7 MINUTES). FOR THE PF SYSTEMS (SISR, APF, MPF, AMPF) WE SHOW THE MEAN (μ) AND THE STANDARD DEVIATION (σ) OVER TEN RUNS.

	FM		AMLt		InfG		Runtime
	μ	σ	μ	σ	μ	σ	[minutes]
SISR	36.8	0.48	31.3	0.40	0.87	0.013	12.5
APF	38.7	0.46	32.8	0.64	0.91	0.014	19.0
MPF	38.8	0.48	32.7	0.90	0.89	0.022	17.8
AMPF	40.8	0.18	35.8	0.45	0.95	0.013	18.1
HMM1	39.6	-	31.7	-	0.87	-	11.1
HMM2	40.5	-	35.1	-	0.94	-	42.1
HMM3	**42.7**	-	**38.7**	-	**1.08**	-	164.0

TABLE IV

BEAT TRACKING RESULTS AND (AVERAGE) RUNTIMES ON THE 1360-SONG DATASET (TOTAL 861.6 MINUTES). FOR THE PF SYSTEMS (SISR, APF, MPF, AMPF) WE SHOW THE MEAN (μ) AND THE STANDARD DEVIATION (σ) OVER TEN RUNS.

	FM		AMLt		InfG		Runtime
	μ	σ	μ	σ	μ	σ	[minutes]
SISR	62.4	0.43	72.3	0.32	1.93	0.0075	74.9
APF	64.6	0.33	74.9	0.35	1.98	0.0068	100.6
MPF	64.9	0.20	75.1	0.27	1.98	0.0059	115.7
AMPF	66.1	0.27	76.6	0.33	2.00	0.0063	120.1
HMM1	62.9	-	69.8	-	1.73	-	69.0
HMM2	65.4	-	76.2	-	1.94	-	264.0
HMM3	**67.4**	-	**79.9**	-	**2.09**	-	1016.4

As a first conclusion, we can see that the four PF schemes improve in accuracy according to their ability to flexibly follow multi-modal distributions, with the proposed AMPF scheme showing superior performance compared to the other three schemes on all datasets and using all metrics. All standard deviations of the PF schemes are moderate, which indicates that their performance can be expected to be reliable throughout individual evaluations. Comparing the best particle filtering inference scheme,

TABLE V

BEAT AND DOWNBEAT TRACKING RESULTS AND AVERAGE RUNTIMES ON THE BALLROOM_TEST DATASET (TOTAL 106.4 MINUTES). FOR THE PF SYSTEMS (SISR, APF, MPF, AMPF) WE SHOW THE MEAN (μ) AND THE STANDARD DEVIATION (σ) OVER TEN RUNS.

	FM		AMLt		InfG		Db-FM		Runtime
	μ	σ	μ	σ	μ	σ	μ	σ	[minutes]
SISR	74.5	0.99	84.8	0.94	2.39	0.034	37.8	2.11	8.7
APF	76.9	0.88	87.5	0.18	2.49	0.017	45.0	1.32	10.9
MPF	82.7	0.70	89.9	0.50	2.52	0.020	53.4	1.19	12.7
AMPF	83.6	0.46	90.5	0.23	2.52	0.012	55.8	1.56	13.7
HMM1	77.5	-	83.0	-	2.11	-	54.9	-	8.1
HMM2	83.2	-	90.2	-	2.49	-	60.6	-	28.9
HMM3	**85.1**	-	**92.1**	-	**2.68**	-	**63.3**	-	111.8

TABLE VI

BEAT AND DOWNBEAT TRACKING RESULTS AND AVERAGE RUNTIMES ON THE BALLROOM_TEST DATASET (TOTAL 106.4 MINUTES) USING EIGHT RHYTHMIC PATTERN STATES. FOR THE PF SYSTEMS (SISR, APF, MPF, AMPF) WE SHOW THE MEAN (μ) AND THE STANDARD DEVIATION (σ) OVER TEN RUNS.

	FM		AMLt		InfG		Db-FM		Runtime
	μ	σ	μ	σ	μ	σ	μ	σ	[minutes]
SISR	66.0	0.94	72.4	1.11	2.23	0.044	39.6	2.14	26.9
APF	70.2	1.26	76.0	1.05	2.37	0.025	46.4	2.07	43.2
MPF	88.3	1.04	90.1	0.52	2.79	0.028	63.9	1.41	43.8
AMPF	89.3	0.54	90.9	0.22	2.78	0.014	67.6	1.22	44.9
HMM1	85.5	-	87.5	-	2.39	-	68.1	-	24.0
HMM2	89.4	-	90.3	-	2.70	-	72.1	-	87.2
HMM3	**90.5**	-	**91.9**	-	**2.90**	-	**73.5**	-	337.3

AMPF, with the three HMM parametrizations, it is apparent that the largest HMM3 slightly outperforms the AMPF on all three datasets. However, this comes at a high price in terms of runtimes, as can be seen from the rightmost columns in Tables III to V. This result was expected, since it confirms that the HMM on a sufficiently dense grid is able to perform accurate inference that cannot be outperformed using approximate methods (such as the PF) using the same underlying model. Comparing the performance of HMM3 and AMPF in the *SMC* and *1360-song* datasets with the performances of other algorithms on

the same datasets [29], it becomes apparent that the HMM3 generally outperforms all other approaches, while the AMPF simply performs as good as the most performing systems. This finding implies that the underlying bar pointer model is a relatively accurate model for meter inference in music in comparison to the state of the art (for the *SMC* and *1360-song* datasets), and the AMPF represents a fast and accurate approximation to the best performance of HMM3. In the following experiment we will evaluate the potential of increasing the number of rhythmic patterns.

B. Experiment 2: Increasing pattern diversity

In the *second experiment* we enlarge the state space by introducing eight style specific rhythmic pattern states into the model, one for each dance style in the *Ballroom* dataset (due to the low number of samples we merged all three Rumba dance styles into one Rumba category). The resulting eight rhythmic patterns are learned on the *Ballroom_train* subset using the dance style labels that come with the data. We compare the beat and downbeat tracking accuracy of the PF and HMM inference methods applied to this enlarged state space in Table VI. For the MPF, AMPF, and HMM methods, and all evaluation measures except AMLt a clear performance improvement over the accuracies depicted in Table V can be seen. The reason for stagnation in the AMLt is easy to explain: The inclusion of more accurate rhythmic patterns leads to less tempo halving or doubling errors, and to less off-beat estimations in Table VI. While such an improvement can be important in certain applications such as transcription or chord estimation, it does not affect AMLt. Furthermore, the SISR and APF seem to have difficulties to handle the enlarged state-space, as indicated by a drastic decrease of beat tracking performance. In contrast, the MPF, AMPF, and HMMs seem to benefit from the more precise model. We can therefore conclude that a more precise modeling of the rhythmic style in a collection (by using a higher number of rhythmic patterns) has the potential to further increase the performance of our model. However, this might be no longer feasible using the HMM with the highest resolution. The HMM3 takes about three times real time to process the data as shown in the rightmost column of Table VI. Therefore, the inference using a model with several rhythmical pattern states marks the point where approximate inference with PFs becomes necessary, at least with the computational power at the time of writing this paper.

VII. Discussion

Our results on the largest available annotated datasets support that the bar pointer model described in Section III is a relatively accurate model for inferring metrical structure from (metered) musical audio signals. Using an exact inference scheme (HMM) in a densely sampled discretized space, we achieve beat

tracking accuracies that outperform those documented for the best state-of-the-art approaches (comparing Tables III and IV with Tables I and II in [29]). The approximate AMPF scheme still achieves accuracies as high as the best state-of-the-art approaches, slightly inferior to the best evaluated HMM. However, good performance with the HMM inference comes at a high price. We need to use HMMs with a large state space (HMM3 in our experiments), which becomes even larger for experiments with several rhythmic pattern states, as in our experiment on the *Ballroom* dataset (Table VI).

Furthermore, our results indicate that modeling several rhythmic patterns improves the performance for music collections with diverse rhythmical content, at least in music with limited expressiveness as investigated in this paper. However, with the resulting enlargement of the discretized state space the usage of a HMM becomes computationally prohibitive and our proposed AMPF is a fast and accurate alternative.

The experiments show that the AMPF scheme (and to some extend also the MPF) handles the degeneracy problem much better than other PF methods because it maintains the diverse multi-modal probability distribution that is crucial for the tracking of multiple tempo modes that occur in music.

In Fig. 8 we demonstrate for an exemplary audio file that the proposed AMPF and MPF, in contrast to the APF and SISR, are able to track the multiple modes of the posterior (light gray regions in Fig. 8) throughout a recording. The figure shows that both SISR and APF concentrate their particles on a few modes of the posterior after only five seconds of the recording (see bottom row of Fig. 8). In contrast, the AMPF is characterized by the most diverse particle distribution throughout a song. This diversity depicted for the example is typical for the approach, and is the reason for the improved performance of the AMPF compared to the other PF schemes.

It is worth to point out that the results reported in this paper might be further improved if the rhythmic patterns were learned from music that is rhythmically more similar to the test set. For instance, this could be achieved by downbeat annotating a smaller subset of a certain style (e.g., the Greek music samples present in the *1360-song* collection), and then attempting to track the meter in the other Greek music samples. Nevertheless, in order to adapt our model to a certain style, a representative set of songs has to be beat and downbeat annotated. However, apart from such annotation work, no changes to the structure of the model or its inference need to be performed.

VIII. Conclusion

In this paper we presented for the first time a particle filtering scheme for beat tracking in music that takes into account multiple modes of the posterior probability, along with a systematic evaluation

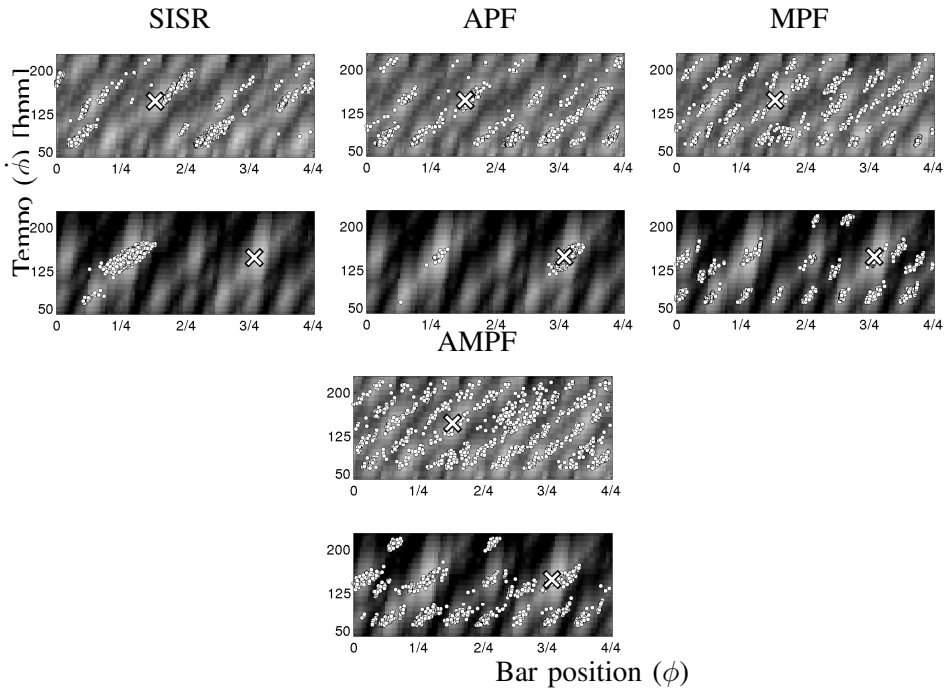

Fig. 8. Particle locations (white dots) within the state-space (tempo and bar position sub-space) after one second (top row) and after five seconds (bottom row) of the song *Lemon Tree* by *Fool's Garden* plotted on top of the (log) posterior probability computed by the HMM. Light gray tones indicate a high posterior probability while dark gray tones indicate a low probability. The cross marks the groundtruth (bar position and tempo for the corresponding time points).

on larger datasets. The comparison with the HMM inference demonstrates its superiority in terms of computational load, while the accuracy of the system is as high as the state-of-the-art in beat tracking. While the presented results prove the applicability of the approach, we need to do further steps in modeling style specific rhythmic patterns in order to better understand the true potential of the method.

Perhaps most importantly, we can claim that the separation of observation model and the hidden variables causes a decoupling of the internal inference from the actual musical sound. This means that the model can potentially be adapted without parameter tweaking or engineering knowledge to new musical styles. By means of this design, the presented method avoids a systematic bias that can result from a hard-coding of music properties into system parameters. It represents a method that can be used for style specific estimation in a straight-forward way by annotating a representative music corpus. Therefore, the system is consistent with the demands of the recent MIR roadmap [19] for systems which are able to incorporate expert knowledge.

REFERENCES

[1] M. Müller, D.P.W. Ellis, A. Klapuri, and G. Richard, "Signal processing for music analysis," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1088–1110, 2011.

[2] S. Dixon, F. Gouyon, and G. Widmer, "Towards characterisation of music via rhythmic patterns," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, 2004.

[3] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1280–1289, 2010.

[4] A. T. Cemgil, H. J. Kappen, and D. Barber, "A generative model for music transcription," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 2, pp. 679–694, 2006.

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

[6] G. Peeters and H. Papadopoulos, "Simultaneous beat and downbeat-tracking using a probabilistic framework: theory and large-scale evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 99, pp. 1–1, 2011.

[7] N. Degara *et al.*, "Reliability-informed beat tracking of musical signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 99, pp. 1–1, 2011.

[8] L.R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[9] S. Roweis and Z. Ghahramani, "A unifying review of linear gaussian models," *Neural computation*, vol. 11, no. 2, pp. 305–345, 1999.

[10] A. Doucet and A.M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," *Handbook of Nonlinear Filtering*, 2009.

[11] A.T. Cemgil and B. Kappen, "Monte carlo methods for tempo tracking and rhythm quantization," *Journal of Artificial Intelligence Research*, vol. 18, no. 1, pp. 45–81, 2003.

[12] S. Hainsworth and M. Macleod, "Particle filtering applied to musical tempo tracking," *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 2385–2395, 2004.

[13] D. Lang and N. de Freitas, "Beat tracking the graphical model way," in *NIPS*, 2004.

[14] N. Whiteley, A.T. Cemgil, and S. Godsill, "Sequential inference of rhythmic structure in musical audio," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Honolulu, 2007, pp. IV–1321.

[15] J. Vermaak, A. Doucet, and P. Pérez, "Maintaining multimodality through mixture tracking," in *Proceedings of the 9th IEEE International Conference on Computer Vision*. IEEE, 2003, pp. 1110–1116.

[16] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American statistical association*, vol. 94, no. 446, pp. 590–599, 1999.

[17] N. Whiteley, A. Cemgil, and S. Godsill, "Bayesian modelling of temporal structure in musical audio," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, Victoria, 2006.

[18] E. Bozdag, "Bias in algorithmic filtering and personalization," *Ethics and Information Technology*, vol. 15, no. 3, pp. 209–227, 2013.

[19] X. Serra *et al.*, *Roadmap for Music Information ReSearch*, Creative Commons BY-NC-ND 3.0 license, 2013.

[20] M. Kolinski, "A cross-cultural approach to metro-rhythmic patterns," *Ethnomusicology*, vol. 17, no. 3, pp. 494–506, 1973.

[21] H. Stobart and I. Cross, "The Andean anacrusis: Rhythmic structure and perception in easter songs of northern Potosí, Bolivia," *British Journal of Ethnomusicology*, vol. 9, no. 2, 2000.

[22] K. Murphy, *Dynamic bayesian networks: representation, inference and learning*, Ph.D. thesis, University of California, Berkeley, 2002.

[23] F. Krebs, S. Böck, and G. Widmer, "Rhythmic pattern modeling for beat and downbeat tracking in musical audio," in *Proc. of the 14th International Conference on Music Information Retrieval (ISMIR)*, Curitiba, 2013.

[24] S. Böck, F. Krebs, and M. Schedl, "Evaluating the online capabilities of onset detection methods," in *Proceedings of the 14th International Conference on Music Information Retrieval (ISMIR)*, Porto, 2012.

[25] S. Godsill, A. Doucet, and M. West, "Maximum a posteriori sequence estimation using monte carlo particle filters," *Annals of the Institute of Statistical Mathematics*, vol. 53, no. 1, pp. 82–96, 2001.

[26] R. Douc and O. Cappé, "Comparison of resampling schemes for particle filtering," in *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA)*. IEEE, 2005, pp. 64–69.

[27] G. Kitagawa, "Monte carlo filter and smoother for non-gaussian nonlinear state space models," *Journal of computational and graphical statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[28] A. Johansen and A. Doucet, "A note on auxiliary particle filters," *Statistics & Probability Letters*, vol. 78, no. 12, pp. 1498–1504, 2008.

[29] A. Holzapfel, M. Davies, J. Zapata, J. Oliveira, and F. Gouyon, "Selective sampling for beat tracking evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 9, pp. 2539–2548, 2012.

[30] F. Gouyon, *A computational approach to rhythm description-Audio features for the computation of rhythm periodicity functions and their use in tempo induction and music content processing*, Ph.D. thesis, Universitat Pompeu Fabra, 2005.

[31] F. Gouyon, S. Dixon, E. Pampalk, and G. Widmer, "Evaluating rhythmic descriptors for musical genre classification," in *Proc. of the AES 25th International Conference*, London, 2004, pp. 196–204.

[32] J.P. Bello *et al.*, "A tutorial on onset detection in music signals," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 1035–1047, 2005.

[33] M. Davies, N. Degara, and M.D. Plumbley, "Evaluation methods for musical audio beat tracking algorithms," *Queen Mary University of London, Tech. Rep. C4DM-09-06*, 2009.

Florian Krebs received the Diploma degree in Electrical Engineering - Audio Engineering from University of Technology and University of Music and Dramatic Arts Graz, Austria in 2010. He is currently a Ph.D. candidate at the Department of Computational Perception of the Johannes Kepler University Linz, Austria. His work focuses on the automatic analysis of music, including onset detection, beat tracking, tempo estimation and expressive performance analysis with interest in probabilistic graphical models.

Andre Holzapfel is currently a post-doctoral researcher at Boğaziçi University, Istanbul, funded by a Marie-Curie IEF grant. In parallel he pursues his second doctoral degree in music at the Center for Advanced Music Studies (MIAM), Istanbul. Before his work in Istanbul he was working as a researcher for the CompMusic project (University Pompeu Fabra), for INESC TEC Porto, and for the Austrian Research Institute for Artificial Intelligence. He obtained his first Ph.D. in Computer Science at the University of Crete. His MIR related research focuses on models and inference schemes for the structure of music, with an emphasis on rhythm. His research in ethnomusicology focuses on music of Crete, and he investigates subjects of the interaction of music and technology. He is a regular performer of Greek Rembetiko music, with his main instruments the Turkish oud and the guitar. He currently directs a documentary movie on amateur Fado in the city of Porto. For further information, refer to `www.rhythmos.org`

Ali Taylan Cemgil received his Ph.D. (2004) from SNN, Radboud University Nijmegen, the Netherlands. Between 2004 and 2008 he worked as a postdoctoral researcher at Amsterdam University and the Signal Processing and Communications Lab., University of Cambridge, UK. He is currently an associate professor of Computer Engineering at Bogazici University, Istanbul, Turkey. He is an associate editor of IEEE SIGNAL PROCESSING LETTERS. His research interests are in Bayesian statistical methods, approximate inference, machine learning and audio signal processing.

Gerhard Widmer is Professor and Head of the Department of Computational Perception at Johannes Kepler University, Linz, Austria, and Head of the Intelligent Music Processing and Machine Learning Group at the Austrian Research Institute for Artificial Intelligence, Vienna, Austria. His research interests include AI, machine learning, and intelligent music processing, and his work is published in a wide range of scientific fields, from AI and machine learning to audio, multimedia, musicology, and music psychology. He is a Fellow of the European Coordinating Committee on Artificial Intelligence (ECCAI), and has been awarded Austria's highest research awards, the START Prize in 1998 and the WITTGENSTEIN Award in 2009, for his work on AI and music.