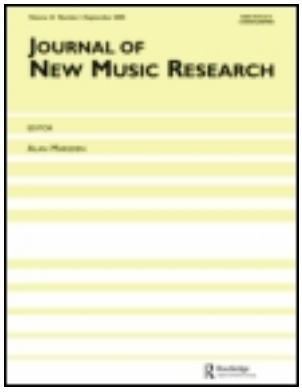

Journal of New Music Research
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/nnmr20

Probabilistic Models for Real-time Acoustic Event Detection with Application to Pitch Tracking

Umut Şimşekli [a] & Ali Taylan Cemgil [a]
[a] Boğaziçi University, Turkey

Available online: 27 Jun 2011

To cite this article: Umut Şimşekli & Ali Taylan Cemgil (2011): Probabilistic Models for Real-time Acoustic Event Detection with Application to Pitch Tracking, Journal of New Music Research, 40:2, 175-185

To link to this article: http://dx.doi.org/10.1080/09298215.2011.573561

Journal of New Music Research
2011, Vol. 40, No. 2, pp. 175–185

Routledge
Taylor & Francis Group

Probabilistic Models for Real-time Acoustic Event Detection with Application to Pitch Tracking

Umut Şimşekli and Ali Taylan Cemgil

Boğaziçi University, Turkey

Abstract

In this paper we present two probabilistic models for real-time acoustic event detection: the Hidden Markov Model and the Change Point Model. We construct the generative models in such a way that each time slice of the audio spectra is generated from a '*spectral template*' which is multiplied by a volume factor. From this point of view, we treat the event detection problem as a template matching problem where the aim is to infer the active template and its volume while the audio data are observed. The novel contribution in this paper is a Change Point Model for real-time template matching using a conditional Poisson observation model. For this model, we develop an exact inference algorithm and an effective approximation schema. We evaluate the models on online monophonic pitch tracking of two low pitched instruments where we focus on the trade-off between the latency and accuracy of the system. The evaluation results suggest favourable features such as quick detection, graceful degradation and an acceptable level of accuracy when compared with a state-of-the-art monophonic pitch tracking algorithm (YIN). We believe that these models provide a flexible and powerful modelling framework for real-time event and pitch detection.

1. Introduction

With the rapid growth of the computational power, real-time computer music systems have become popular in both artistic and entertainment applications. In order for the interaction to be fluent, these systems require quick response in real-time while providing a comprehensive analysis of music in order to be accurate. Therefore accurate and flexible event detection methods are needed.

In this study we propose and evaluate two probabilistic models for real-time detection of acoustic events. These events in question can be different notes played by a harmonic instrument, percussive sounds that are generated by humans (i.e. hand clapping, finger snapping) or percussive instrument sounds (i.e. cymbals, membraphones), and so on. The main concern of the work is reducing the detection latency without compromising the detection quality. Here, the term latency is defined as the time difference between the true event onset and the time that the system has computed its estimate. Clearly, the more data are accumulated the more accurate the estimates should be. However, we wish earliest detection as possible to reduce the latency.

From our point of view, there are two reasons for a real-time acoustic event detection method to have latency: one is intrinsic and the other one is extrinsic. The intrinsic reason is that the method cannot estimate the onset accurately because it has not accumulated enough data yet. This is in some sense a theoretical limit of a given model or method, independent of the speed of a particular computer running the algorithm. The second, extrinsic reason is the computational burden; here latency occurs due to poor implementation or other practicalities such as delays of audio device drivers. We assume that for an algorithm that performs a constant amount of computation for each additional sample, these latter extrinsic reasons can be virtually eliminated by using more powerful computers and careful programming. Hence, in our work we focus only on the intrinsic properties of an event detector and study in detail the latency/accuracy trade-off. In other words, for a

Correspondence: Umut Şimşekli, Department of Computer Engineering, Boğaziçi University, 34342 Bebek, Istanbul, Turkey. E-mail: {umut.simsekli,taylan.cemgil}@boun.edu.tr

DOI: 10.1080/09298215.2011.573561

particular model, we aim to estimate the lower bound of the processing delay, as a function of accuracy.

The advantage of the proposed framework is that it can be applied to several types of applications, relevant for acoustic processing. In this study we tested the framework on real-time monophonic pitch tracking where we used recordings of two low pitched instruments: a tuba and a bass guitar. This is considered challenging since estimating low pitches in shortest time is intrinsically a difficult problem due to the longer wavelengths and the 'blurring' in the low frequency spectrum. We conduct our experiments on the electric bass guitar and tuba recordings of the RWC Musical Instrument Sound Database. Encouraged by the simulation results, we have implemented the framework as a plug-in for popular real-time signal processing environments Pure Data and Max/MSP, suggesting the applicability of the methods in practice.

1.1 Related work

Pitch tracking is one of the most studied topics in the computer music field since it lies at the centre of many applications. It is widely used in phonetics, speech coding, music information retrieval, music transcription, and interactive musical performance systems. It is also used as a pre-processing step in more comprehensive music analysis applications such as chord recognition systems.

Many pitch tracking methods have been presented in the literature; indeed the algorithms are so numerous that it is very difficult, if not impossible to give a complete summary. The main trends can be summarized as algorithmic and model based approaches. Puckette, Apel, and Zicarelli (1998) presented a maximum-likelihood pitch detector and developed an object called 'fiddle~' for the real-time signal processing systems PD and Max/MSP. Klapuri (2008) proposed an auditory model based fundamental frequency estimator for polyphonic music and speech signals. As another algorithmic approach, Saito, Kameoka, Takahashi, Nishimoto, and Sagayama (2008) presented the Specmurt analysis technique, where the pitch estimation is achieved by deconvolution of the audio signal after transforming it in the specmurt domain. Assuming that each sound in a polyphonic signal has exactly the same harmonic structure pattern in the log-frequency domain, the specmurt method describes the overall shape of the audio spectrum as the convolution of a fundamental frequency pattern and the common harmonic structure pattern.

Model based approaches combine elements of subspace techniques or probabilistic models. In a recent review, Christensen, Stoica, Jakobsson, and Holdt Jensen (2008) propose and evaluate four statistical signal processing methods for single and multi-pitch estimation. Yeh, Roebel, and Chang (2008) proposed a multiple pitch estimation method which is composed of two parts. In the first part, they determined the number of sources (i.e. polyphony) and the related fundamental frequencies by a frame-by-frame basis. Then, they utilized a Hidden Markov Model in order to refine the estimation that was obtained from the first part of their method. Ryynänen and Klapuri (2008) proposed a method for the automatic transcription of melody, bass line, and chords in polyphonic music. The method incorporates both heuristic and model-based techniques, such as pitch salience estimation, acoustic modelling, and musicological modelling, where the Hidden Markov Models are utilized for acoustic and musicological modelling. Cemgil (2004) also proposed generative models for both monophonic and polyphonic music transcription.

Recently, nonnegative matrix factorization (NMF) methods have become popular for various audio processing applications and have found its place in music transcription. Different types of NMF models with different assumptions and inference schemes have been proposed and evaluated on polyphonic music analysis (Cont, 2006; Vincent, Bertin, & Badeau, 2008; Févotte, Bertin, & Durrieu, 2009; Peeling, Cemgil, & Godsill, 2010). For a more comprehensive overview of different pitch estimation/detection methods, the curious reader is referred to Klapuri and Davy (2006).

The current approach combines a NMF-like model with the change point approach introduced first in Şimşekli (2010) and Şimşekli and Cemgil (2010), which reported preliminary results. A Hidden Markov Model for online recognition of percussive events is reported in Şimşekli, Jylhä, Erkut, and Cemgil (in press). In this study, we compare a similar Hidden Markov Model and a novel improved Change Point Model to the problem of quick onset detection and pitch tracking and compare their performances on monophonic audio, in terms of detection quality and estimation delay.

The novel contributions of this paper are as follows.

- We develop a novel conditionally Poisson Change Point Model for real-time template matching.
- For the Change Point Model, we develop an exact inference algorithm, an effective approximation schema and a training algorithm.
- We introduce a detailed evaluation methodology that focuses on the trade-off between the intrinsic latency and detection accuracy.
- We report detailed simulation results for a bass guitar and tuba.

The rest of the paper is organized as follows. In Section 2, the required technical background is provided. The probabilistic models are presented in Section 3. The inference and training schemes are presented in Sections 4 and 5. We report our results in Section 6 and finally, Section 7 concludes this paper.

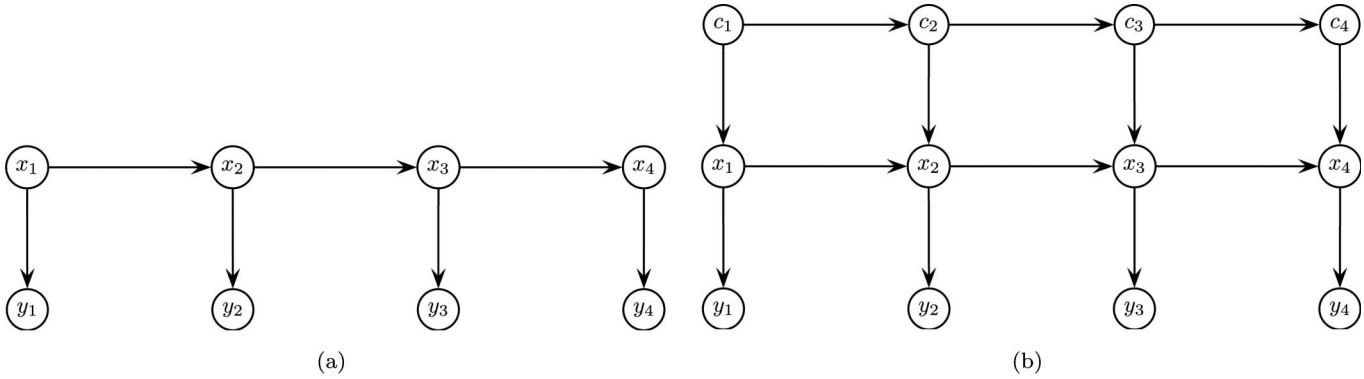

Fig. 1. Graphical model of (a) a Hidden Markov Model and (b) a Change Point Model. x_τ represent the latent variables, y_τ represent the observations, and c_τ represent the binary switch variables. These graphs visualize the conditional independence structure between the random variables and allows the joint distribution to be rewritten by utilizing Equation 1. In the model, the nodes correspond to probability distributions of model variables, and edges to their conditional dependencies.

2. Technical background

Audio processing can be seen as time-series processing where a time-series is defined as a sequence of observations which are measured at an increasing set of time points (usually uniformly spaced). In this study, we will be dealing with two probabilistic models for time-series modelling: the Hidden Markov Model and the Change Point Model.

2.1 Hidden Markov Model

A Hidden Markov Model (HMM) is a statistical model which is basically a partially observed Markov chain (Cappé, Moulines, & Ryden, 2005). At each time point τ, we have a latent state x_τ, that is not directly observable. Instead, we observe a related random variable y_τ. The goal is to estimate the hidden states given the observations.

In Figure 1(a), we show a so-called 'graphical model' of a standard HMM (Barber & Cemgil, 2010), which provides an intuitive way to represent the conditional independence structure of the probabilistic model. In the graphical model, the nodes correspond to probability distributions of model variables, and edges to their conditional dependencies. The joint distribution can be rewritten by making use of the directed acyclic graph:

$$p(x_{1:T}, y_{1:T}) = \prod_{\tau=1}^{T} p(x_\tau|\mathrm{pa}(x_\tau))p(y_\tau|\mathrm{pa}(y_\tau)), \qquad (1)$$

where pa (χ) denotes the *parent nodes* of χ. As can be seen from the graphical model, the hidden state variable at time τ depends only on the state variable at time $\tau-1$. This is called the Markov property[1].

$$p(x_\tau|x_{1:\tau-1}) = p(x_\tau|x_{\tau-1}). \qquad (2)$$

Similarly, the observation at time τ depends only on the state variable at time τ,

$$p(y_\tau|y_{1:\tau-1}, x_{1:\tau}) = p(y_\tau|x_\tau). \qquad (3)$$

In a HMM, the probability distribution in Equation 2 is called the *state transition model* and the distribution in Equation 3 is called the *observation model*. The HMM is called *homogeneous* if the state transition and the observation models do not depend on time index τ, which is our case in this study.

2.2 Change Point Model

In the classic time-series models, the underlying latent process is assumed to be either discrete (i.e. Hidden Markov Model) or continuous (i.e. Kalman Filter). These kinds of models have been shown to be successful in many problems from various research fields. However, in some cases selecting the underlying process either discrete or continuous would not be sufficient. Thanks to the increase in the computational power and the development in the state-of-the-art inference methods, we are able to construct more complex statistical models such as the Change Point Models (see Barber & Cemgil 2010, and references herein).

A Change Point Model (CPM) is a switching state space model where the variables have a special structure. In a CPM, we have two latent variables: the **discrete** switch variable c_τ and the **continuous** variable x_τ. While the switch variable is off ($c_\tau = 0$), x_τ follows the pre-defined structure that depends on $x_{\tau-1}$. On the other hand, at the time when the switch variable is on ($c_\tau = 1$),

[1]Note that we use MATLAB's colon operator syntax in which (1: T) is equivalent to [1, 2, 3,...,T] and $x_{1:T} \equiv \{x_1, x_2, \ldots, x_T\}$.

x_τ is reset to a new value independent from the previous values.

In this model, the switch variables c_τ form a Markov chain. Besides, conditioned on c_τ, x_τ also form a Markov chain. The graphical model representation of a CPM is shown in Figure 1(b).

3. Probabilistic modelling of acoustic events

In this section, we infer a predefined set of pitch labels from streaming audio data. We construct two probabilistic models that relate a latent event label to the actual audio recording. The audio signal is subdivided into frames and represented by the magnitude spectrum of the frames which is calculated with discrete Fourier transform. We define $x_{v,\tau}$ as the magnitude spectrum of the audio data with frequency index v and time index τ, where $v \in \{1, 2, \ldots, F\}$ and $\tau \in \{1, 2, \ldots, T\}$. Here, F is the number of frequency bins and T is the number of time frames.

For each time frame τ, we define an indicator variable r_τ on a discrete state space D_r, which determines the label we are interested in. In our case D_r consists of note labels such as {C4, C#4, D4, D#4, ..., C6}. The indicator variables r_τ are hidden since we do not observe them directly.

In our models, the main idea is that each event has a certain characteristic spectral shape which is rendered by a specific volume. The spectral shapes that we denote as *spectral templates* are denoted by $t_{v,i}$. The v index is again the frequency index and the index i indicates the pitch labels. Here, i takes values between 1 and I, where I is the number of different spectral templates. The volume variables v_τ define the overall amplitude factor, by which the whole template is multiplied. An overall sketch of the model is given in Figure 2.

3.1 Hidden Markov Model

Hidden Markov Models have been widely studied in various types of applications such as audio processing, natural language processing, and bioinformatics. Like in several computer music applications, HMMs have also been used in pitch tracking applications (Raphael, 2002; Orio & Sette, 2003).

We define the probabilistic model as follows:

$$\begin{aligned} r_0 &\sim p(r_0), \\ r_\tau | r_{\tau-1} &\sim p(r_\tau | r_{\tau-1}), \\ v_\tau &\sim \mathcal{G}(v_\tau; a_v, b_v), \\ x_{v,\tau} | v_\tau, r_\tau &\sim \prod_{i=1}^{I} \mathcal{PO}(x_{v,\tau}; t_{v,i} v_\tau)^{[r_\tau = i]}. \end{aligned} \tag{4}$$

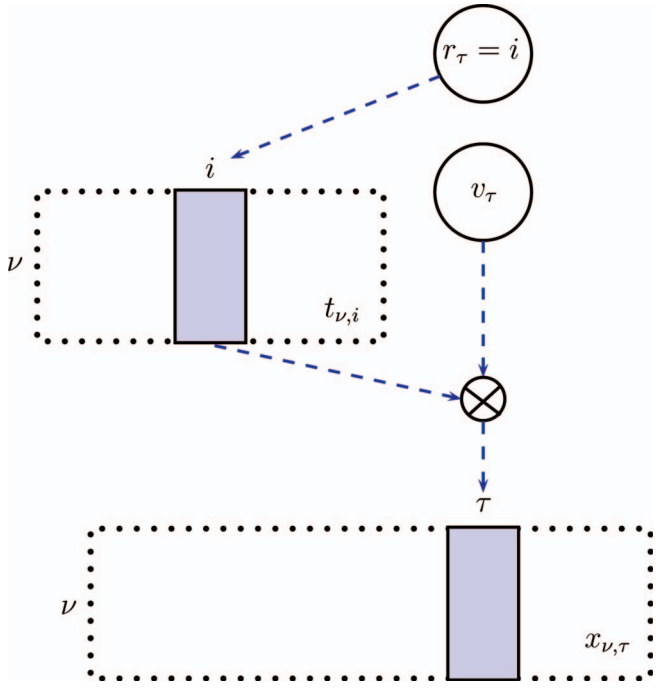

Fig. 2. The block diagram of the probabilistic models. The indicator variables, r_τ choose which template to be used. The chosen template is multiplied by the volume parameter v_τ in order to obtain the magnitude spectrum, $x_{v,\tau}$.

Here $[x] = 1$ if x is true, $[x] = 0$ otherwise and the symbols G and $\mathcal{PO}$ represent the Gamma and the Poisson distributions respectively, where

$$\begin{aligned} \mathcal{G}(x; a, b) &= \exp\left((a-1)\log x - bx - \log\Gamma(a) + a\log(b)\right) \\ \mathcal{PO}(x; \lambda) &= \exp\left(x\log\lambda - \lambda - \log\Gamma(x+1)\right), \end{aligned} \tag{5}$$

where Γ is the Gamma function. For an integer x, we have $\Gamma(x+1) = x!$, the factorial function.

In some recent work on polyphonic pitch tracking, NMF models are widely used (Vincent et al., 2008; Févotte et al., 2009). One popular approach uses the KL divergence as the error metric when fitting a model to a spectrogram. It is shown in Cemgil (2009), that this choice is equivalent to a Poisson observation model. Since our probabilistic models are conceptually similar to NMF models, we choose a Poisson distribution as the observation model. We also choose Gamma prior on v_τ to preserve conjugacy and make use of the scaling property of the Gamma distribution. Other choices, such as Gaussians are also possible but are not investigated further in this paper.

We choose a Markovian prior on the indicator variables, r_τ which means r_τ depends only on $r_{\tau-1}$. Following a similar approach as in Orio and Sette (2003),

we use three states to represent a note: one state for the attack part, one for the sustain part, and one for the release part. We also use a single state in order to represent silence. Figure 3(a) shows the graphical model of the HMM.

In this probabilistic model we can integrate out analytically the volume variables, v_τ. It is easy to check that once we do this, provided the templates $t_{v,i}$ are already known, the model reduces to a standard HMM with a Compound Poisson observation model (Şimşekli, 2010).

The observation model assumes that the subsequent frames are conditionally independent from each other given the latent indicators r_τ. Hence, to conform with this assumption, we calculate the spectra $x_{v,\tau}$ on nonoverlapping frames. In practice, one could also compute the spectrum using overlapping frames but then the conditional independence assumption would not be exactly valid.

3.2 Change Point Model

In addition to the HMM, in the Change Point Model (CPM), the volume parameter v_τ has a specific structure which depends on $v_{\tau-1}$ (i.e. staying constant, monotonically increasing or decreasing, etc.). But at certain unknown times, it jumps to a new value independently from $v_{\tau-1}$. We call these times '*change points*'. The occurrence of a change point is determined by the binary switch variable c_τ. If c_τ is on, in other words if $c_\tau = 1$, then a change point has occurred at time τ.

The formal definition of the generative model is given below:

$$\begin{aligned}
v_0 &\sim \mathcal{G}(v_o; a_0, b_0),\\
r_0 &\sim p(r_0),\\
c_\tau &\sim \mathcal{BE}(c_\tau; w),\\
r_\tau|c_\tau, r_{\tau-1} &\sim \begin{cases} p_0(r_\tau|r_{\tau-1}), & c_\tau = 0,\\ p_1(r_\tau|r_{\tau-1}), & c_\tau = 1,\end{cases}\\
v_\tau|c_\tau, r_\tau, v_{\tau-1} &\sim \begin{cases} \delta(v_\tau - \theta(r_\tau)v_{\tau-1}), & c_\tau = 0,\\ \mathcal{G}(v_\tau; a_v, b_v), & c_\tau = 1,\end{cases}\\
x_{v,\tau}|v_\tau, r_\tau &\sim \prod_{i=1}^{I} \mathcal{PO}(x_{v,\tau}; t_{v,i}v_\tau)^{[r_\tau = i]}.
\end{aligned} \tag{6}$$

Here, $\delta(x)$ is the Kronecker delta function which is defined by $\delta(x) = 1$ when $x = 0$, and $\delta(x) = 0$ elsewhere. The symbol $\mathcal{BE}$ represents the Bernoulli distribution, where

$$\mathcal{BE}(x; \omega) = \exp\left(x \log \omega + (1 - x) \log (1 - \omega)\right). \tag{7}$$

The graphical representation of the probabilistic model is given in Figure 3(b).

The $\theta(\cdot)$ function determines the specific structure of the volume variables where, $\theta(r_\tau) \in \{\theta_a, \theta_s, \theta_r\}$. Here θ_a, θ_s, and θ_r correspond to the attack, sustain, and release parts of a note respectively. $\theta(r_\tau)$ gives flexibility to the CPM since we can adjust it with respect to the instrument whose sound would be processed (i.e. we can select $\theta_a = \theta_s = \theta_r = 1$ for woodwind instruments by assuming the volume of a single note would stay approximately

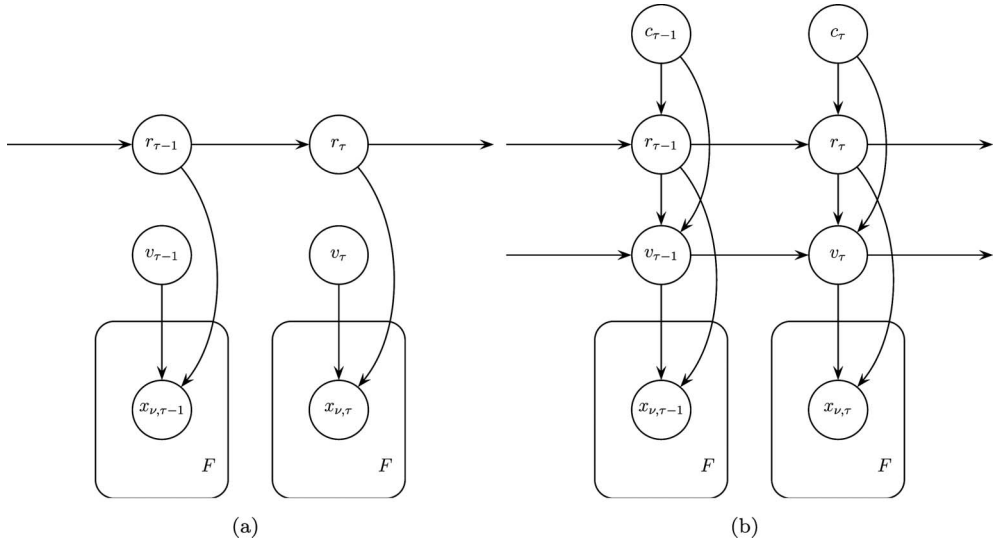

Fig. 3. Graphical model of the (a) HMM and (b) CPM. Note that we use the plate notation for the observed variables where F distinct nodes (i.e. $x_{v,\tau}$ where $v \in \{1, \ldots, F\}$) are grouped and represented as a single node in the graphical model. In this case, F is the number or frequency bins.

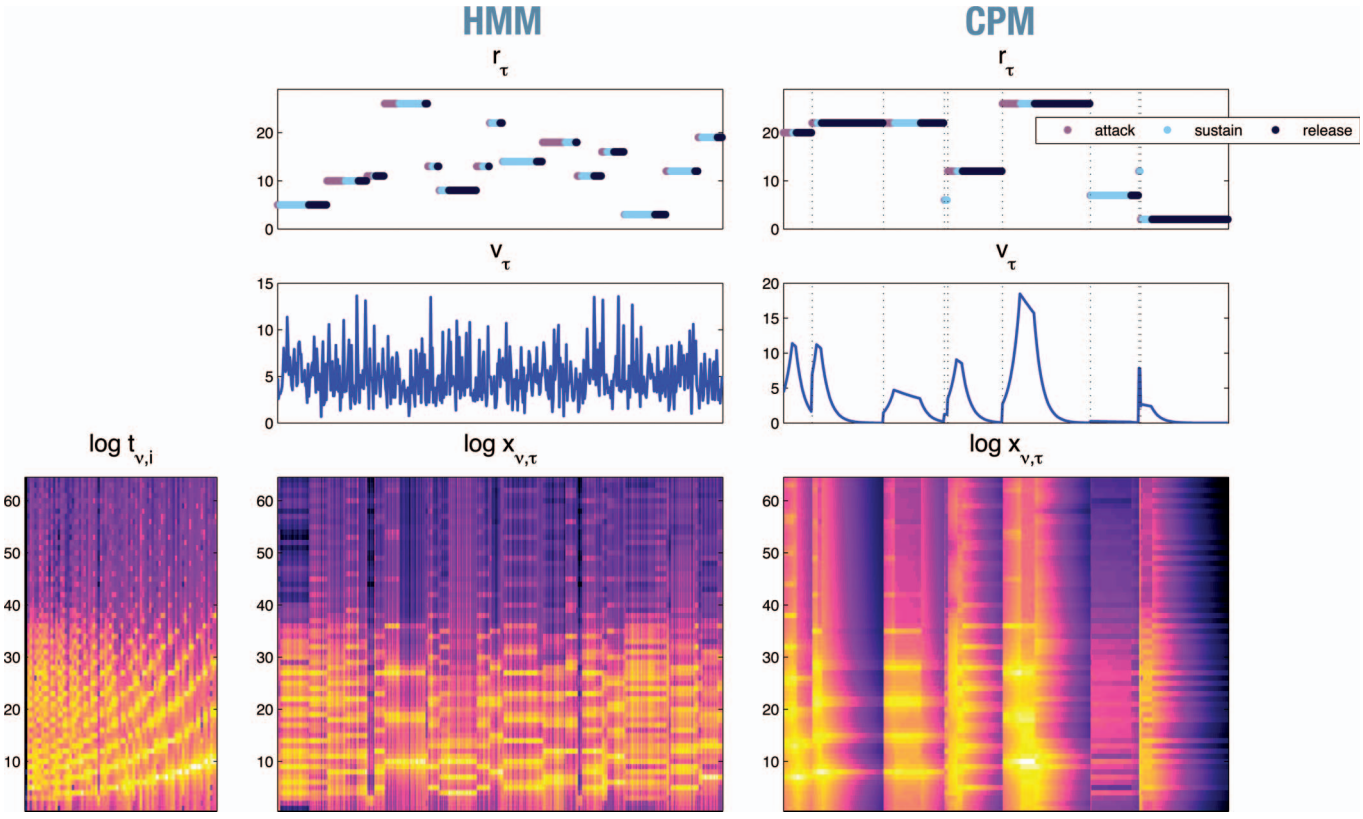

Fig. 4. Spectral templates of a tuba and synthetic data generated from the HMM and CPM. The topmost figures show a realization of the indicator variables r_τ and the second topmost figures show a realization of the volume variables v_τ. The bottommost figures show the spectral templates and the audio spectra that are generated from the HMM and CPM respectively. The dashed lines represent the points where the change points occur. It can be observed that the CPM is more natural in terms of modelling an audio spectrum.

constant). Figure 4 visualizes example templates and synthetic data which are generated from the CPM.

Note that a similar Change Point Model for music transcription has been presented by Cemgil, Kappen, and Barber (2006). That model is similar to our model in terms of the dependence structure of the latent variables; however, it has a sinusoidal model-based observation model which makes heavy assumptions about the harmonic structure of audio. As opposed to that model, the proposed Change Point Model is linked to the audio signal by a template-based observation model which enables the model to be used in several applications.

4. Inference

Inference is a fundamental issue in probabilistic modelling where we ask the question 'what can be the hidden variables as we have some observations?' (Cappé et al., 2005). For online processing, we are interested in the computation of the so-called filtering density: $p(r_\tau|x_{1:F,1:\tau})$, that reflects the information about the current state r_τ given all the observations $x_{1:F,1:\tau}$ so far. The filtering density can be computed online, however the estimates that can be obtained from it are not necessarily very accurate as future observations are not accounted for.

An inherently better estimate can be obtained from the so-called fixed lag smoothing density, if we can afford to wait a few steps more. In other words, in order to estimate r_τ, if we accumulate L more observations, at time $\tau+L$, we can compute the distribution $p(r_\tau|x_{1:F,1:\tau+L})$ and estimate r_τ via:

$$r_\tau^* = \arg\max_{r_\tau} \; p(r_{1:\tau+L}|x_{1:F,1:\tau+L}). \tag{8}$$

Here, * denotes the optimality, L is a specified lag and it determines the trade-off between the accuracy and the latency. By accumulating a few observations from the future, the detection at a specific frame can be eventually improved at the cost of introducing a slight latency. Therefore, we have to fine-tune this parameter in order to have balance in the latency–accuracy trade-off. In the following subsections, we will explain the inference schemes of the HMM and the CPM respectively for calculation of these quantities.

As a reference to compare against, we will compute an inherent batch quantity: the most likely label

trajectory given all the observations, the so-called Viterbi path

$$r^*_{1:T} = \arg\max_{r_{1:T}} \; p(r_{1:T}|x_{1:F,1:T}). \tag{9}$$

This quantity requires that we accumulate all data before estimation and should give a high accuracy at the cost of very long latency.

4.1 Hidden Markov Model

The goal of inference in the HMM is computing the filtering and the (fixed-lag) smoothing distributions and the (fixed-lag) Viterbi path which are defined at the beginning of Section 4. In a standard HMM, these quantities can be computed by the well-known forward–backward algorithm where the forward (α) and the backward (β) messages are defined as:

$$\begin{aligned}\alpha_\tau(r_\tau) &= p(r_\tau, x_{1:F,1:\tau}),\\ \beta_\tau(r_\tau) &= p(x_{1:F,\tau+1:T}|r_\tau).\end{aligned} \tag{10}$$

We can compute these messages via the following recursions:

$$\begin{aligned}\alpha_\tau(r_\tau) &= p(x_{1:F,\tau}|r_\tau)\sum_{r_{\tau-1}} p(r_\tau|r_{\tau-1})\alpha_{\tau-1}(r_{\tau-1}),\\ \beta_\tau(r_\tau) &= \sum_{r_{\tau+1}} p(r_{\tau+1}|r_\tau)p(x_{1:F,\tau+1}|r_{\tau+1})\beta_{\tau+1}(r_{\tau+1}).\end{aligned} \tag{11}$$

Here, $\alpha_0(r_0) = p(r_0)$ and $\beta_T(r_T) = 1$ (Barber & Cemgil, 2010). Once these messages are computed, the smoothing distribution can be computed easily by multiplying the forward and backward messages as

$$p(r_\tau|x_{1:F,1:T}) \propto \alpha_\tau(r_\tau)\beta_\tau(r_\tau), \tag{12}$$

where $\propto$ denotes the proportionality up to a multiplicative constant. Besides, the Viterbi path is obtained by replacing the *summations* over r_τ by *maximization* in the forward recursion.

The good news about this model is that we can integrate out analytically the volume variables, v_τ. Hence, given that the templates $t_{v,i}$ are already known, the model reduces to a standard Hidden Markov Model with a Compound Poisson observation model as shown below (see Şimşekli 2010 for details):

$$\begin{aligned}&p(x_{1:F,\tau}|r_\tau = i)\\ &= \int dv_\tau \, \exp\left(\sum_{v=1}^{F} \log \mathcal{PO}(x_{v,\tau}; v_\tau t_{v,i}) + \log \mathcal{G}(v_\tau; a_v, b_v)\right)\\ &= \frac{\Gamma\left(\sum_{v=1}^{F} x_{v,\tau} + a_v\right)}{\Gamma(a_v)\prod_{v=1}^{F}\Gamma(x_{v,\tau}+1)} \; \frac{b_v^{a_v}\prod_{v=1}^{F} t_{v,i}^{x_{v,\tau}}}{\left(\sum_{v=1}^{F} t_{v,i} + b_v\right)^{\sum_{v=1}^{F} x_{v,\tau}+a_v}}.\end{aligned} \tag{13}$$

Since we have standard HMM from now on, we can run the forward algorithm in order to compute the filtering density or fixed-lag versions with a few backward steps. Also we can estimate the most probable state sequence by running the Viterbi algorithm. A benefit of having a standard HMM is that the inference algorithm can be made to run very fast. This allows the inference scheme to be implemented in real-time without any approximation (Alpaydın, 2004).

4.2 Change Point Model

While making inference on the CPM, our task is finding the posterior probability of the change point variables c_τ, indicator variables r_τ, and the volume variables v_τ. If v_τ were discrete, then the CPM would reduce to an ordinary HMM with a latent state that is an element of the set $D_c \times D_r \times D_v$, where D_c, D_r, and D_v denote the state spaces of c_τ, r_τ, and v_τ respectively. However in our case v_τ is continuous, an exact forward–backward algorithm cannot be implemented in general. This is due to the fact that the prediction density $p(c_\tau, r_\tau, v_\tau \,|\, x_{1:F,\tau})$ needs to be computed by integrating over $v_{\tau-1}$ and summing over $c_{\tau-1}$ and $r_{\tau-1}$. Unfortunately, the summation over discrete variables $c_{\tau-1}$ and $r_{\tau-1}$ does not 'simplify' the prediction density. This density becomes a (Gamma) mixture model where each mixture component corresponds to a possible setting of the discrete variables and the number of mixture components grows linearly with increasing τ. Whilst this is still manageable for short sequences, exact inference becomes impractical for online processing as the algorithm is requiring increasingly more computation. In order to eliminate this problem, an approximate inference scheme is utilized where we systematically prune low probability components of the mixture. Figure 5 illustrates the inference scheme and the pruning procedure. In the figure, the solid arrows represent the case of the change point, and the dashed arrows represent the opposite case. The shaded area illustrates the pruning procedure where the Gamma potentials with lowest mixture coefficients are pruned and the number of the mixture components are guaranteed to be constant. The detailed derivation of the forward–backward algorithm for the CPM as well as a more detailed analysis of the pruning strategy can be found in Şimşekli (2010).

4.2.1 Marginal Viterbi path

The marginal Viterbi path is defined as:

$$(c^*_{1:T}, r^*_{1:T}) = \arg\max_{c_{1:T}, r_{1:T}} \int_{v_{1:T}} p(x_{1:F,1:T}, c_{1:T}, r_{1:T}, v_{1:T}).$$

In the CPM, replacing the summations over r_τ and c_τ by maximization can be problematic since maximization and

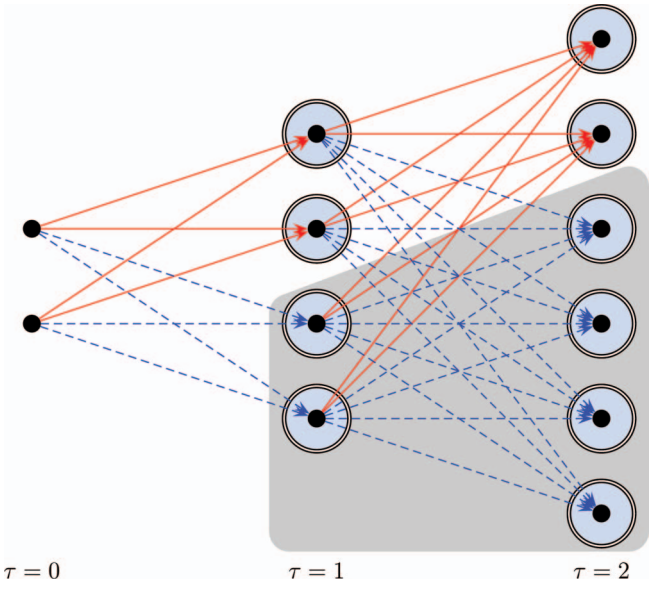

Fig. 5. Visualization of the forward and the Viterbi algorithm for the CPM. Here, the number of templates, I is chosen to be 2. The small dots represent the Gamma potentials. For the forward procedure, the big circles represent the *sum* operator that sums the mixture coefficient of the Gamma potentials. For the Viterbi procedure, we replace the *sum* operator with the *max* operator which selects the Gamma potential that has the maximum mixture coefficient.

integration do not commute. We integrate over the hidden variables v_τ first, in other words we compute the mixture coefficients of the Gamma potentials. Then we select the maximum of them. We call this path 'marginal', since in order to achieve the exact Viterbi path, we should have also replaced the integration over v_τ by maximization in Equation 14. Fortunately, for this model, we are able to compute the exact marginal distribution of r_τ and c_τ, $p(c_{1:T},r_{1:T}|x_{1:F,1:T})$, and the exact marginal Viterbi path (Cemgil et al., 2006). Intuitively, the resulting algorithm is no different from smoothing. We merely replace the *sum* operators with *max* operators in Figure 5. For a detailed discussion, see Şimşekli (2010).

5. Training and parameter learning

So far, we have constructed the inference algorithms with the assumption that the templates $t_{v,i}$ are known. In this section, we describe how the spectral templates $t_{v,i}$ can be estimated from data by using an Expectation–Maximization (EM) algorithm. This algorithm iteratively maximizes the log-likelihood as follows:

E-step:

$$\begin{aligned}&\text{HMM}: q(r_{1:T}, v_{1:T})^{(n)} = p(r_{1:T}, v_{1:T}|x_{1:F,1:T}, t_{1:F,1:I}^{(n-1)}),\\ &\text{CPM}: q(c_{1:T}, r_{1:T}, v_{1:T})^{(n)} = p(c_{1:T}, r_{1:T}, v_{1:T}|x_{1:F,1:T}, t_{1:F,1:I}^{(n-1)}),\end{aligned} \tag{14}$$

M-step:

$$\begin{aligned}&\text{HMM}: t_{1:F,1:I}^{(n)} = \arg\max_{t_{1:F,1:I}} \left\langle \log p(r_{1:T}, v_{1:T}, x_{1:F,1:T}|t_{1:F,1:I}) \right\rangle_{q(r_{1:T}, v_{1:T})^{(n)}},\\ &\text{CPM}: t_{1:F,1:I}^{(n)} = \arg\max_{t_{1:F,1:I}} \left\langle \log p(c_{1:T}, r_{1:T}, v_{1:T}, x_{1:F,1:T}|t_{1:F,1:I}) \right\rangle_{q(c_{1:T},r_{1:T},v_{1:T})^{(n)}},\end{aligned} \tag{15}$$

where $\langle f(x)\rangle_{p(x)} = \int p(x)\, f(x)\, dx$ is the expectation of the function $f(x)$ with respect to $p(x)$.

In the E-step, we compute the posterior distributions of r_τ and v_τ for the HMM and the posterior distributions of c_τ, r_τ, and v_τ for the CPM. These quantities can be computed via the methods which we described in Subsections 4.1 and 4.2 for the HMM and the CPM respectively. In the M-step, which is a fixed point equation, we want to find the $t_{v,i}$ that maximize the likelihood; the solution is given as:

$$t_{v,i}^{(n)} = \frac{\sum_{\tau=1}^{T} \langle [r_\tau = i] \rangle^{(n)} x_{v,\tau}}{\sum_{\tau=1}^{T} \langle [r_\tau = i] v_\tau \rangle^{(n)}}. \tag{16}$$

Intuitively, we can interpret this result as the weighted average of the normalized audio spectra with respect to v_τ.

6. Results

In order to evaluate the performance of the probabilistic models on pitch tracking, we have conducted several experiments. As mentioned earlier, in this study we focus on the monophonic pitch tracking of low-pitched instruments. We have measured and compared the accuracy and the latency of the models by varying the amount of lag in the fixed-lag Viterbi algorithm, which is decribed in Section 4.

In our experiments we used the electric bass guitar and tuba recordings of the RWC Musical Instrument Sound Database. We first trained the templates offline, and then we tested our models by utilizing the previously learned templates.

At the training step, we ran the EM algorithm which we described in Section 5, in order to estimate the spectral templates. For each note we used a short isolated recording. On the whole, we use 28 recordings for bass guitar (from E2 to G4) and 27 recordings for tuba (from F2 to G4). The HMM's training phase lasts approximately 30 s and the CPM's lasts approximately 2 min on a standard computer.

At the testing step, we rendered monophonic MIDI files to audio by using the samples from RWC recordings. The total duration of the test files are approximately 5 min. At the evaluation step, we compared our estimates with the ground truth which is obtained from the MIDI file. In both our models we used 46 ms long frames at 44.1 kHz sampling rate.

From our point of view, the main trade-off of these pitch tracking models is between the latency and the accuracy. We can increase the accuracy by accumulating the data, in other words increasing the latency. However after some point the pitch tracking system would be useless due to the high latency. Hence we tried to find reasonable latency and accuracy by adjusting the 'lag' parameter of the fixed-lag Viterbi path which is defined in Equation 8.

As evaluation metrics, we used the recall rate, the precision rate, the speed factor and the note onset latency. The recall and precision rate, and latency is defined in Table 1.

The evaluation results of the probabilistic models are shown in Figure 6. It can be observed that enlarging the lag yields higher precision and recall rates; however, this also increases the overall latency of the system at the same time. Therefore, we notice that a lag of around 135 ms seems reasonable for both models: we obtain 94.5% precision and 94% recall with the HMM and 99.5% precision and 94% recall with the CPM. Besides, increasing the lag does not affect the results after some degree and the fixed-lag results converge to offline results after ≈250 ms.

In Figure 7, we show the performance of the CPM on two different instruments, bass guitar and tuba. Since the sound structures of a plucked string instrument and a brass instrument are different, the performance would differ from one instrument to another as expected. From the figure, it can be observed that the bass guitar fits better than the tuba to this model. This is not surprising since the CPM captures the physical properties of a plucked string instrument better than a brass instrument.

We also compared the performance of our models with the well-known YIN algorithm (Cheveigné & Kawahara, 2002). Despite the fact that YIN is a general purpose method, we compared our results with the YIN's, since YIN is accepted as a standard method for monophonic pitch tracking. We used the *aubio* implementation and tuned the onset threshold parameter. The results are shown in Table 2.

Table 1. Definition of the evaluation metrics. Note that latency is computed *without* considering the label of the estimate.

precision	$\frac{\text{num. of correct notes}}{\text{num. of transcribed notes}}$
recall	$\frac{\text{num. of correct notes}}{\text{num. of true notes}}$
onset latency	estimated onset time - true onset time

6.1 Real-time implementation

Encouraged by the simulation results, we implemented the HMM in real-time. We first implemented the framework by using MATLAB's 'Data Acquisition Toolbox'. Despite that this toolbox neither works on any operating systems other than 32 bit MS Windows, nor supports low-latency ASIO drivers, we achieved good results. However, in order to have a faster and portable implementation, by using the **boost** C++ libraries and **flext** C++ development layer, we also implemented the framework as a plug-in for popular real-time environments Pure Data and Max/MSP. For details of the implementation, the curious

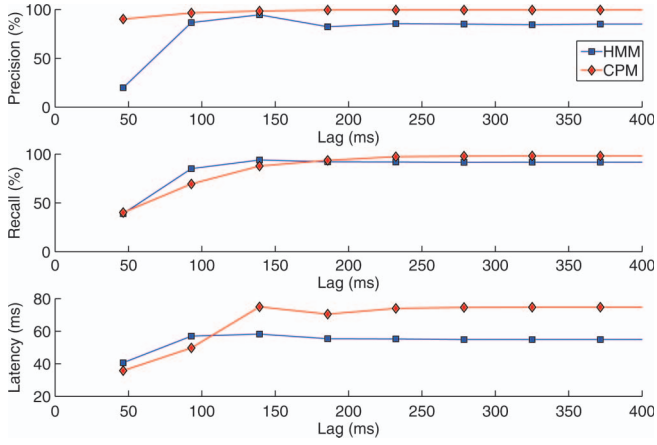

Fig. 6. The average performance of the probabilistic models on low-pitched audio. The graphics show the precision and the recall rate, and latency from top to bottom. Note that the total latency of the system is the sum of the lag and the latency at the note onsets (y-axis in the bottommost figure).

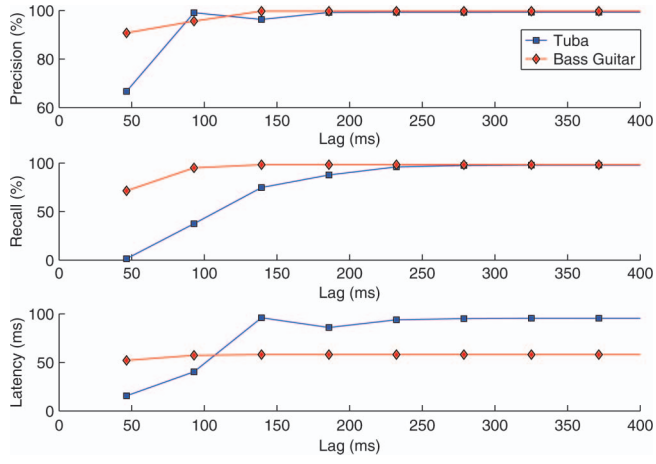

Fig. 7. The average performance of the CPM on different instruments.

Table 2. The comparison of our models with the YIN algorithm. The speed factor is defined as the ratio between the running time of the method and the duration of the test file and is a cpu-dependent metric which would be lower in faster computers. It is observed that the CPM performs better than the others. Moreover, the HMM would also be advantageous due to its cheaper computational needs and lower latency.

	Recall (%)	Precision (%)	Onset latency (ms)	Speed factor
YIN	43.43	9.40	58.74	1.33
HMM	91.72	85.02	54.89	0.02
CPM	98.06	99.50	74.74	0.05

reader is referred to Şimşekli et al. (in press). The HMM object is available at http://www.cmpe.boun.edu.tr/~umut/eventtracking.

7. Discussion and conclusions

In this study we presented and compared two probabilistic models for real-time acoustic event detection. In our models, it is assumed that each event has a certain characteristic spectral shape which we call the *spectral template*. The generative models were constructed in such a way that each time slice of the audio spectra is generated from one of these spectral templates multiplied by a volume factor. From this point of view, we treated the event detection problem as a template matching problem where the aim is to infer which template is active and what the volume is as we observe the audio data.

The main focus on this work was the trade-off between latency and accuracy of the pitch tracking system. We conducted several experiments in order to find reasonable accuracy and latency. We evaluated the performance of our models by computing the most-likely paths that were obtained via filtering or fixed-lag smoothing distributions. The evaluation was held on monophonic bass guitar and tuba recordings with respect to four evaluation metrics. We also compared the results with the YIN algorithm and obtained better results.

The proposed models are also extensible to more complicated scenarios such as polyphony. This can be done by using factorial models (Cemgil, 2006) or using hierarchical NMF models where in this case r_τ and v_τ would be vectors instead of scalars. This kind of extension requires more complex inference schemes, and we aim to investigate more powerful inference methods for such models.

As mentioned earlier, our framework can also be used for several audio processing applications such as percussive event detection. Thanks to the flexibility of the framework, for percussive event detection, we only need to replace the spectral templates of the notes with spectral templates of the percussive events. Şimşekli et al. (in press) presented the evaluation results of the HMM on several percussive events.

We believe that the CPM provides a flexible and powerful modelling framework for real-time event and pitch detection.

Acknowledgements

We would like to thank the reviewers for helpful comments and suggestions. This work is funded by The Scientific and Technical Research Council of Turkey (TÜBİTAK) grant number 110E292, project 'Bayesian matrix and tensor factorisations (BAYTEN)'. The work of Umut Şimşekli is supported by the PhD scholarship (2211) from TÜBİTAK.

References

Alpaydın, E. (2004). *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*. Cambridge, MA: MIT Press.

Barber, D., & Cemgil, A.T. (2010). Graphical models for time series. *IEEE Signal Processing Magazine (Special issue on graphical models)*, *27*(27), 18–28.

Cappé, O., Moulines, E., & Ryden, T. (2005). *Inference in Hidden Markov Models* (Springer Series in Statistics). Secaucus, NJ: Springer-Verlag New York, Inc.

Cemgil, A.T. (2004). *Bayesian music transcription* (PhD thesis). Radboud University of Nijmegen, the Netherlands.

Cemgil, A.T. (2006). Sequential inference for factorial changepoint models. In *Nonlinear Statistical Signal Processing Workshop*, Cambridge, UK, pp. 203–206.

Cemgil, A.T. (2009). Bayesian inference in non-negative matrix factorisation models. *Computational Intelligence and Neuroscience*. Article ID 785152.

Cemgil, A.T., Kappen, H.J., & Barber, D. (2006). A generative model for music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(2), 679–694.

Cheveigné, A. de, & Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. *Journal of Acoustical Society of America*, *111*, 1917–1930.

Christensen, M.G., Stoica, P., Jakobsson, A., & Holdt Jensen, S. (2008). Multi-pitch estimation. *Signal Processing*, *88*(4), 972–983.

Cont, A. (2006). Realtime multiple pitch observation using sparse non-negative constraints. In *ISMIR 2006 – 7th International Conference on Music Information Retrieval*, Victoria, Canada, pp. 206–211.

Févotte, C., Bertin, N., & Durrieu, J.-L. (2009). Nonnegative matrix factorization with the Itakura-Saito divergence. With application to music analysis. *Neural Computation*, *21*(3), 793–830.

Klapuri, A. (2008). Multipitch analysis of polyphonic music and speech signals using an auditory model. *IEEE Transactions on Audio, Speech & Language Processing*, *16*(2), 255–266.

Klapuri, A., & Davy, M. (2006). *Signal Processing Methods for Music Transcription*. New York: Springer.

Orio, N., & Sette, M.S. (2003). An HMM-based pitch tracker for audio queries. In *ISMIR 2003 – 4th International Symposium on Music Information Retrieval*, Baltimore, MD, USA, pp. 249–250.

Peeling, P.H., Cemgil, A.T., & Godsill, S.J. (2010). Generative spectrogram factorization models for polyphonic piano transcription. *Transactions on Audio, Speech and Language Processing*, *18*(3), 519–527.

Puckette, M., Apel, T., & Zicarelli, D. (1998). Real-time audio analysis tools for Pd and MSP. In *Proceedings of International Computer Music Conference (ICMC)*, Ann Arbor, MI, USA, pp. 109–112.

Raphael, C. (2002). Automatic transcription of piano music. In *ISMIR 2002 – 3rd International Symposium on Music Information Retrieval*, Paris, France, pp. 15–19.

Ryynänen, M.P., & Klapuri, A.P. (2008). Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, *32*(3), 72–86.

Saito, S., Kameoka, H., Takahashi, K., Nishimoto, T., & Sagayama, S. (2008). Specmurt analysis of polyphonic music signals. *IEEE Transactions on Audio, Speech & Language Processing*, *16*(3), 639–650.

Şimşekli, U. (2010). *Bayesian methods for real-time pitch tracking* (Master's thesis). Boğaziçi University, Turkey.

Şimşekli, U., & Cemgil, A.T. (2010). A comparison of probabilistic models for online pitch tracking. In *Proceedings of the 7th Sound and Music Computing Conference (SMC)*, Barcelona, Spain, pp. 502–509.

Şimşekli, U., Jylhä, A., Erkut, C., & Cemgil, A.T. (in press). Real-time recognition of percussive sounds by a model-based method. *EURASIP Journal on Advances in Signal Processing*, *2011*.

Vincent, E., Bertin, N., & Badeau, R. (2008). Harmonic and inharmonic nonnegative matrix factorization for polyphonic pitch transcription. In *ICASSP'08 IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, USA, pp. 109–112.

Yeh, C., Roebel, A., & Chang, W.-C. (2008). Mutiple-F0 estimation for MIREX 2008. In *9th International Conference on Music Information Retrieval (ISMIR'08)*, Philadelphia, PA, USA.