Synthesizing Musical Accompaniments with Bayesian Belief Networks

Christopher Raphael
Department of Mathematics and Statistics
University of Massachusetts at Amherst
Amherst, MA 01003-4515
raphael@math.umass.edu

Nov. 22, 2000
revised May 22, 2001

Abstract

This paper discusses recent work in creating a computer program that plays the role of a sensitive musical accompanist. An accompanist must synthesize a number of different sources of information including a real-time analysis of the soloist's acoustic signal, an understanding of the timing relationships represented in the musical score, the interpretation of the soloist learned through rehearsals, and musical constraints on the way in which the accompaniment can be played. A probabilistic framework is presented in which all of these knowledge sources can be represented and learned from actual data. This model then forms the basis of an approach to musical accompaniment using the machinery of Bayesian Belief Networks. A demonstration is provided from J.S. Bach's Cantata 12.

1 Introduction

Our objective is to build a system that provides musical accompaniment for a live player performing a non-improvisatory composition for soloist and accompaniment. The input to the system comes from a microphone focussed on the live player. The program performs a real-time analysis of this signal and occasionally decides, in response to what it has "heard," that it is time to play an accompaniment note. When this happens MIDI message is sent to an electronic musical instrument or sound module which performs the actual sound synthesis. This signal is then directed to an amplifier which drives a loud speaker. When our efforts succeed, the accompaniment directed by the program will actually respond to the soloist's playing and will *follow* the soloist. Other efforts in this area include [Dannenberg 84], [Bloch & Dannenberg 85], [Dannenberg & Mukaino 88] and [Vercoe & Puckette 85],[Vercoe 84]. A more complete account of this research is discussed in the paper "A Probabilistic Expert System for Automatic Musical Accompaniment" available at http://fafner.math.umass.edu.

Two processes: "Listen" and "Play" form the heart of our system. The Listen process uses a hidden Markov model to track the evolution of the soloist's position within the score; this work

is described in detail in [Raphael 99] and will not be discussed here. When the Listen process determines that a solo note has occurred it communicates this information to the Play process; these messages come with variable latency due to the degree of local ambiguity of the acoustic signal. The task of the Play process is to play the accompaniment using the note onset times provided by the Listen process, as well as several other knowledge sources. It is the Play process that we discuss in this work.

What knowledge must an accompanist consider? Certainly we need the information prescribed in the printed score such as the idealized pitches and relative note lengths as well as the real-time analysis of the acoustic signal provided by Listen. However, the accompanist must consider more than this. As the piece of music is rehearsed the soloist demonstrates a musical interpretation and somehow the accompanist must learn and incorporate these ideas into its own playing. But "copying" the soloist's interpretation, by itself, is not enough. The accompanist must have its own musical agenda if the accompaniment is to be musically satisfying.

While all of these knowledge sources must be simultaneously considered in performing the accompaniment, there is an additional constraint our system must respect Since musical performance is inherently a real-time process, our system must also function in real time. This means that, whatever methods, models, and algorithms are employed, they must be designed with computational feasibility in mind.

2 A Probabilistic Model

We develop here a model that captures fundamental aspects of the time evolution of the solo and accompaniment parts as well as their interaction. Our current treatment does not model *dynamics*, (i.e. loudness information), however our model could be extended to include this facet of musical performance. Our model is given as a joint probability distribution on a collection of hundreds of random variables, which represent tangible aspects of musical performance such as local tempo and note onset time for both parts. Some of these are directly observable, e.g. measured onset time, and some are not, e.g. local tempo. One cannot reasonably expect to represent and train such a distribution without making assumptions that simplify the structure of the joint distribution. We will make a number of musically informed conditional independence assumptions that limit the degree of interaction among the random variables we model. In particular, we represent our probabilistic model as a distribution on a directed acyclic graph (DAG), thus explicitly modeling the dependency among the variables. Having done this, our model is amenable to the techniques of Bayesian Belief Networks (BBNs). We further assume that all of the variables in our model are jointly Gaussian; this modeling assumption is necessary to ensure the computationally feasibility of our method. In particular, we will show in Section 3 how the BBN machinery forms the backbone of our real-time playing algorithm, as well as the learning algorithm of Section 4.

2.1 The Solo Model

Here we propose a simple random model that expresses the evolution of a generic solo part. In particular, we wish to develop a model that describes the joint distribution on the actual times at which the solo notes occur as well as an evolving "tempo process."

Suppose our solo part is composed of a sequence of $N+1$ notes and rests comprising what we will call the "solo events." The solo events have written lengths $l_0, \ldots, l_N$; we express these written lengths in terms of measures, so, for example, a quarter note in 6/8 time would have a length of

1/3. Let the vector $(t_n, s_n)^t$ represent the "state" of the n^{th} event where t_n is the event's onset time, in seconds, and s_n is the event's local tempo, expressed in seconds per measure. Consider the model that describes the evolution of these state vectors,

$$\begin{pmatrix} t_n \\ s_n \end{pmatrix} = \begin{pmatrix} 1 & l_{n-1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_{n-1} \\ s_{n-1} \end{pmatrix} + \begin{pmatrix} \tau_n \\ \sigma_n \end{pmatrix} \tag{1}$$

for $n = 1, \ldots, N$ where $\tau_1, \ldots, \tau_{n-1}$ and $\sigma_1, \ldots, \sigma_{n-1}$ are random variables. If the $\{(\tau_n, \sigma_n)^t\}$ vectors are removed from Eqn. 1, then the tempo process is constant, $s_0 = s_1 = \ldots = s_N$, and the note durations, $\{t_{n+1} - t_n\}$ are proportional to the written note lengths $\{l_n\}$ as in a march. The inclusion of these random vectors allows two kinds of flexibility into our model. A positive value of σ_n means that the measure size is increasing at the n^{th} note, hence we have a local *ritardando* or a slowing down of the tempo. A negative value corresponds to a decrease in measure size signifying a local *accelerando* or speeding up of the tempo . A positive value of τ_n describes a place in which the actual note length is longer than would be predicted by the local measure size, s_n, yet no tempo change occurs. In musical terms this would be a stretch or *agagic accent.* Similarly, a negative value of τ_n would compress a note.

Our model is expressed more compactly and more completely as

$$x_n^{\text{solo}} = A_{n-1}^{\text{solo}} x_{n-1}^{\text{solo}} + \xi_n^{\text{solo}} \tag{2}$$

for $n = 1, \ldots, N$ where

$$A_n^{\text{solo}} = \begin{pmatrix} 1 & l_n \\ 0 & 1 \end{pmatrix} \tag{3}$$

$$x_n^{\text{solo}} = \begin{pmatrix} t_n \\ s_n \end{pmatrix} \tag{4}$$

$$\xi_n^{\text{solo}} = \begin{pmatrix} \tau_n \\ \sigma_n \end{pmatrix} \tag{5}$$

and where x_0^{solo} and $\xi_1^{\text{solo}}, \ldots, \xi_N^{\text{solo}}$ are mutually independent Gaussian random vectors with

$$x_0^{\text{solo}} \sim N(\mu_0^{\text{solo}}, \Sigma_0^{\text{solo}}) \tag{6}$$

and

$$\xi_n^{\text{solo}} \sim N(\mu_n^{\text{solo}}, \Sigma_n^{\text{solo}}) \tag{7}$$

$n = 1, \ldots, N$. Learning the soloist's interpretation will be formulated as parameter estimation for the $\{\mu_n^{\text{solo}}, \Sigma_n^{\text{solo}}\}$ parameters in Section 4.

2.2 Incorporating the Observations

Recall that the "Listen" process analyzes the acoustic signal generated by the soloist and generates a sequence of real numbers corresponding to the estimated onset times for the solo notes. We now notate these by $x_0^{\text{obs}}, \ldots, x_N^{\text{obs}}$, where x_n^{obs} is the estimated onset time for the n^{th} solo note. How do these times relate to the idealized state process $x_0^{\text{solo}}, \ldots, x_{n-1}^{\text{solo}}$? Our model assumes that

$$x_n^{\text{obs}} = A_n^{\text{obs}} x_n^{\text{solo}} + \xi_n^{\text{obs}} \tag{8}$$

where

$$A_n^{\text{obs}} = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

and

$$\xi_n^{\text{obs}} \sim N(0, \sigma_n^{\text{obs}})$$

with $\xi_0^{\text{obs}}, \ldots, \xi_N^{\text{obs}}$ mutually independent and independent of x_0^{solo} and $\xi_1^{\text{solo}}, \ldots, \xi_N^{\text{solo}}$ as well. Thus, the $x_0^{\text{obs}}, \ldots, x_N^{\text{obs}}$ are "noisy" observations of the idealized note onset times $t_0, \ldots, t_N$. There are at least two factors that contribute to the discrepancy between t_n and x_n^{obs}. First of all, our measurement process is imperfect due to quantization errors and occasional misfirings of the Listen process. Second, there is always a certain amount of imprecision in the playing of a musical instrument, so we might assume that the ideally desired times are never perfectly realized. These effects are summarized in the distributions of the $\{\xi_n^{\text{obs}}\}$. The dependence of the observation variance on n is meant to account for the variable performance of the Listen process. This is because the detection of some events, such as rearticulations and rest onsets are more error prone than others. It seems reasonable to assume there is no systematic bias to these errors, hence the $\{\xi_n^{\text{obs}}\}$ are assumed to be 0 mean.

2.3 Representing Probabilities on Graphs

Let $G = (\Gamma, \Lambda)$ be a directed acyclic graph (DAG) where Γ is a collection of graph nodes and Λ is a collection of directed edges. Let x be a random vector that is partitioned so that each component of x is associated with a node, $\gamma \in \Gamma$. If $\Delta \subseteq \Gamma$, we write x_Δ for the random vector composed of components of x associated with members of Δ. By convention, we write x_γ instead of the more correct $x_{\{\gamma\}}$ and we write x_Γ instead of x when we wish to emphasize that we are referring to the entire collection of random variables.

A probability distribution for $x = x_\Gamma$ *admits recursive factorization* with respect to G if it has a density, $f(x)$, that can be represented as

$$f(x) = \prod_{\gamma \in \Gamma} f_\gamma(x_\gamma | x_{\text{pa}(\gamma)}) \tag{9}$$

where the parents of x_γ, $\text{pa}(x_\gamma)$, are nodes in Γ that have edges leading to γ and f_γ is the conditional density for x_γ given its parents $x_{\text{pa}(\gamma)}$ [Lauritzen 96]. For our case we assume

$$x_\gamma = A_\gamma x_{\text{pa}(\gamma)} + \xi_\gamma \tag{10}$$

where the $\{\xi_\gamma\}$ are independent random vectors with $\xi_\gamma \sim N(\mu_\gamma, \Sigma_\gamma)$ and the $\{A_\gamma\}$, $\{\mu_\gamma\}$, and $\{\Sigma_\gamma\}$ are chosen to replicate the conditional distribution $f_\gamma(x_\gamma | x_{\text{pa}(\gamma)})$. Thus x_Γ has a joint Gaussian distribution.

As an example, the joint distribution of $\{x_0^{\text{solo}}, \ldots, x_N^{\text{solo}}\}$ and $\{x_0^{\text{obs}}, \ldots, x_N^{\text{obs}}\}$, was defined through Eqns. 2 and 8, both of the same form as Eqn. 10. Thus the joint distribution of these random variables respects the graph of Fig. 1 with the indicated association between random variables and nodes. In this graph every node, except the node corresponding to x_0^{solo} has a single "parent" node.

2.4 Adding the Accompaniment

Here we outline our representation of the conditional distribution of the accompaniment variables given the solo variables. Rather than completely specifying this conditional distribution, (which

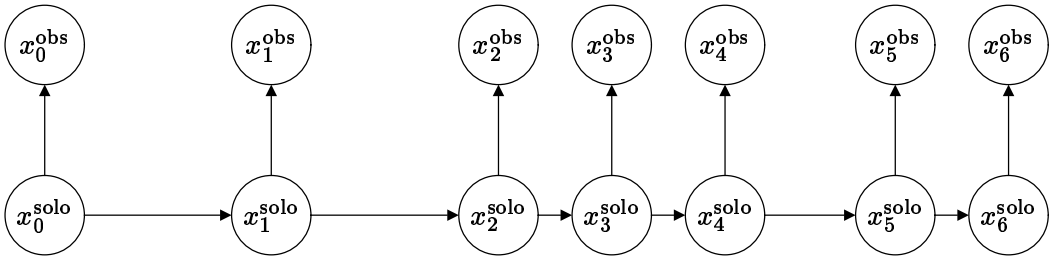

Figure 1: A graphical representation of the joint distribution on $x_0^{\text{solo}} \ldots, x_N^{\text{solo}}$ and $x_0^{\text{obs}} \ldots, x_N^{\text{obs}}$. The horizontal placement of nodes is proportional to their onset times in *musical* time (measures). We follow this convention in all subsequent graphs.

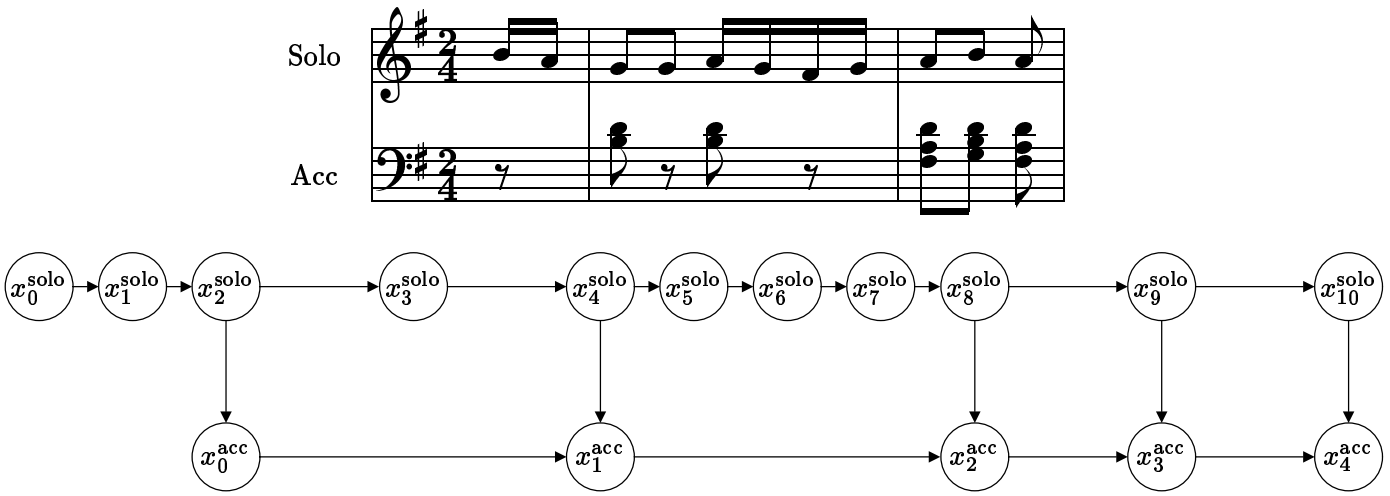

Figure 2: *Top:* A musical example in which each accompaniment event is coincident with a solo event. *Bottom:* A graphical representation of the conditional dependency structure

we eventually intend to learn from examples in future work), we focus here on the conditional independence assumptions we build into the model. As before, these can be expressed through a graph structure.

The prevailing musical wisdom tells us that the accompanist should "follow" the soloist, although, of course, the reality is much more complicated. In practice the role of leader can be exchanged freely the course of the piece. Furthermore, there are times when a musician's role cannot accurately be characterized as follower *or* leader. Still the view of soloist as leader is certainly a reasonable one much of the time, and we feel that this notion must figure into the probabilistic modeling of the relationship between solo and accompaniment.

How do we model this notion probabilistically? Consider the musical example in the top of Figure 2. In this example each accompaniment event occurs at the same musical time as a solo event; all things being equal, we would like these coincident notes to occur at the same *actual* time. We have learned from experience, however, that the accompaniment must take more than synchronicity into account if it is to be musically satisfying. In particular, the accompaniment must have a disposition or tendency that guides it toward a musically plausible performance. For instance, tempo changes over fast notes should, in general, be gradual. Without such musical constraints the accompaniment sounds chaotic and confusing. The two objectives of synchronicity

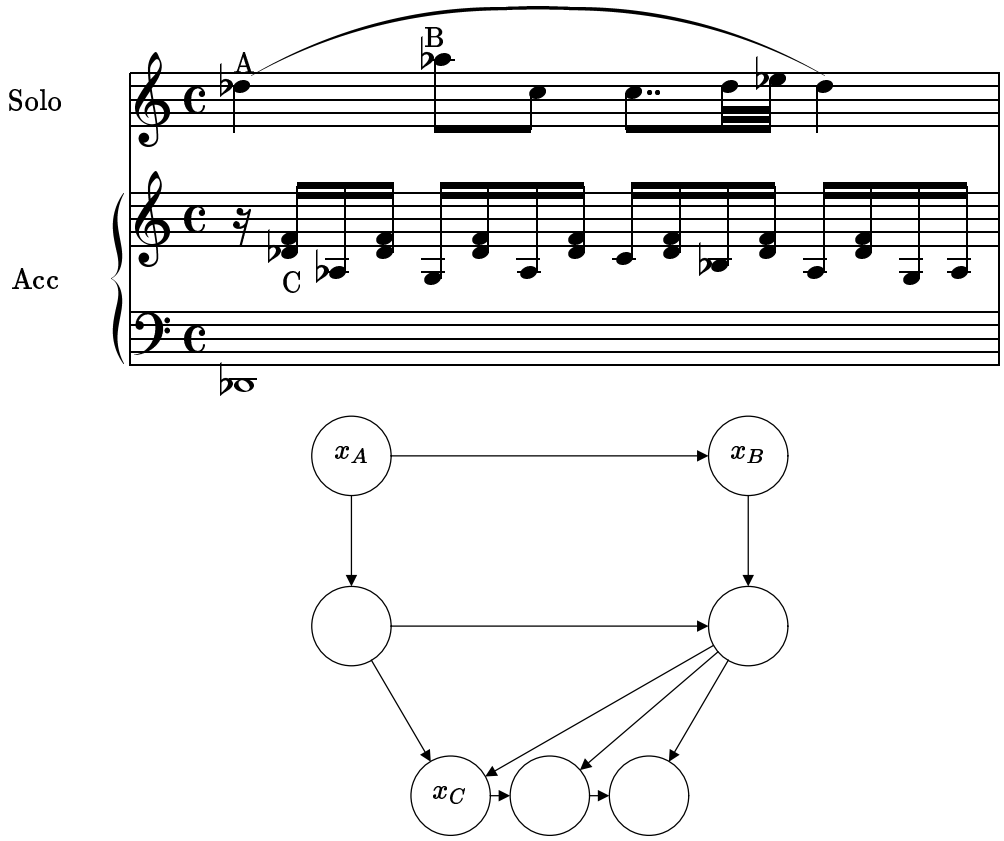

Figure 3: *Top:* A musical excerpt beginning with four accompaniment events evenly spaced over the first solo note. *Bottom:* Our graphical representation of the dependency structure.

and internal consistency are occasionally in opposition to one another. Our goal is to mediate between between these two competing objectives.

Consider a situation in which each accompaniment event is coincident with a solo event and denote the solo and accompaniment state process by $x_0^{\text{solo}}, \ldots, x_N^{\text{solo}}$ and $x_0^{\text{acc}}, \ldots, x_M^{\text{acc}}$. We model the asymmetric roles of the two parts in the following way. We assume that the solo part evolves according to the model of Eqn. 2 where each solo state depends probabilistically only on the preceding solo state. In contrast, we assume that each accompaniment state depends not only on the previous accompaniment state (internal consistency), but also on the concurrent solo state (synchronicity). This dependency structure is illustrated by the graph in the bottom of Figure 2. One might say that the solo process evolves as if it doesn't "hear" the accompaniment, but accompaniment does "hear" the soloist. This is how we model the notion of "following."

Now consider the musical situation in the top of Figure 3 in which there are several accompaniment events between a pair of accompaniment events that coincide with the solo notes marked A and B. We model this situation as in the bottom panel of Fig. 3 in which we make explicit the dependence of x_C on *both* x_A and x_B.

These ideas form the basis of our modeling of the accompaniment variables. A complete graph showing the dependency structure on all of the variables of our model is shown in Figure 4. In this figure the top and bottom layers of random variables correspond to the estimated note onsets times provided by listen and the actual accompaniment event times. The accompaniment event times, which we denote by $x_0^{\text{out}}, \ldots, x_M^{\text{out}}$, depend deterministically on their parent nodes. These

two layers are the only directly observable variables in our model. the remaining nodes comprise a collection of hidden variables that explain the dependency structure between these two observable layers.

3 Playing the Accompaniment

Our method for playing the accompaniment is, in principle, quite simple. At any time, t, during the performance we know the estimated solo event times already detected by Listen: $x_0^{\text{obs}}, \ldots, x_{n(t)}^{\text{obs}}$ and the accompaniment events already played: $x_0^{\text{out}}, \ldots, x_{m(t)}^{\text{out}}$. Figure 5 shows these currently observed variables with solid circles, while the currently unobserved vectors are represented with open circles. Given this information, we can compute the conditional distribution on the time of the next unplayed accompaniment event $x_{m(t)+1}^{\text{out}}$. Note that this distribution is influenced by all of the knowledge sources previously mentioned: the written note lengths of the score, the rhythmic interpretation of the soloist, the estimated onset times from Listen, and the musical constraints modeled by the practice room distribution. We then schedule $x_{m(t)+1}^{\text{out}}$ for a time that is consistent with this distribution. An obvious choice would schedule $x_{m(t)+1}^{\text{out}}$ to sound at the conditional mean of this distribution. This scheduling computation takes place every time we receive new information, i.e. every time Listen detects an event and every time an accompaniment event is played. In the case of a solo event detection, this amounts to rescheduling the currently pending accompaniment event; in the case of a played accompaniment event, this amounts to initializing the next accompaniment note event.

In light of the above, the description of our playing algorithm will be complete once we have described a means of computing posterior distributions on model variables, given the observation of other model variables. The literature on Bayesian Belief Networks, for example [Spiegelhalter et. al. 93], [Lauritzen 96], [Jensen 96], addresses this problem thoroughly and we utilize that theory here. In particular we incorporate Lauritzen's specialization of the BBN theory to the case of Gaussian distributions [Lauritzen 92]. We do not discuss this computation here, except to mention that it is accomplished through the "message passing" algorithm and note that the computations involved are feasible in real time.

4 Training the Model

A basic tenet of our approach holds that, for our accompaniment, as with human musicians, rehearsal is indispensable to successful performance. In the rehearsal phase the accompaniment will learn aspects of the soloist's interpretation whose knowledge is essential to satisfying interaction between soloist and accompaniment. We discuss here a fully automatic method for learning the parameters that model this interaction using the EM algorithm; EM is an iterative technique that estimates model parameters by incrementally improving the likelihood of the observed data under the model parameters. The basic idea we use for training our BBN is thoroughly developed for the case of discrete variables in [Lauritzen 95], and is briefly sketched for the joint Gaussian case which applies to our situation.

The probabilistic model for the entire collection of random variables, x_Γ, presented in Section 2 is described in terms of a large number of independent random vectors, $\{\xi_\gamma\}$, whose distributions characterize aspects of the joint distribution on x_Γ. In principle, our training procedure can be employed to estimate any or all of the parameters governing the $\{\xi_\gamma\}$ vectors. In practice, we

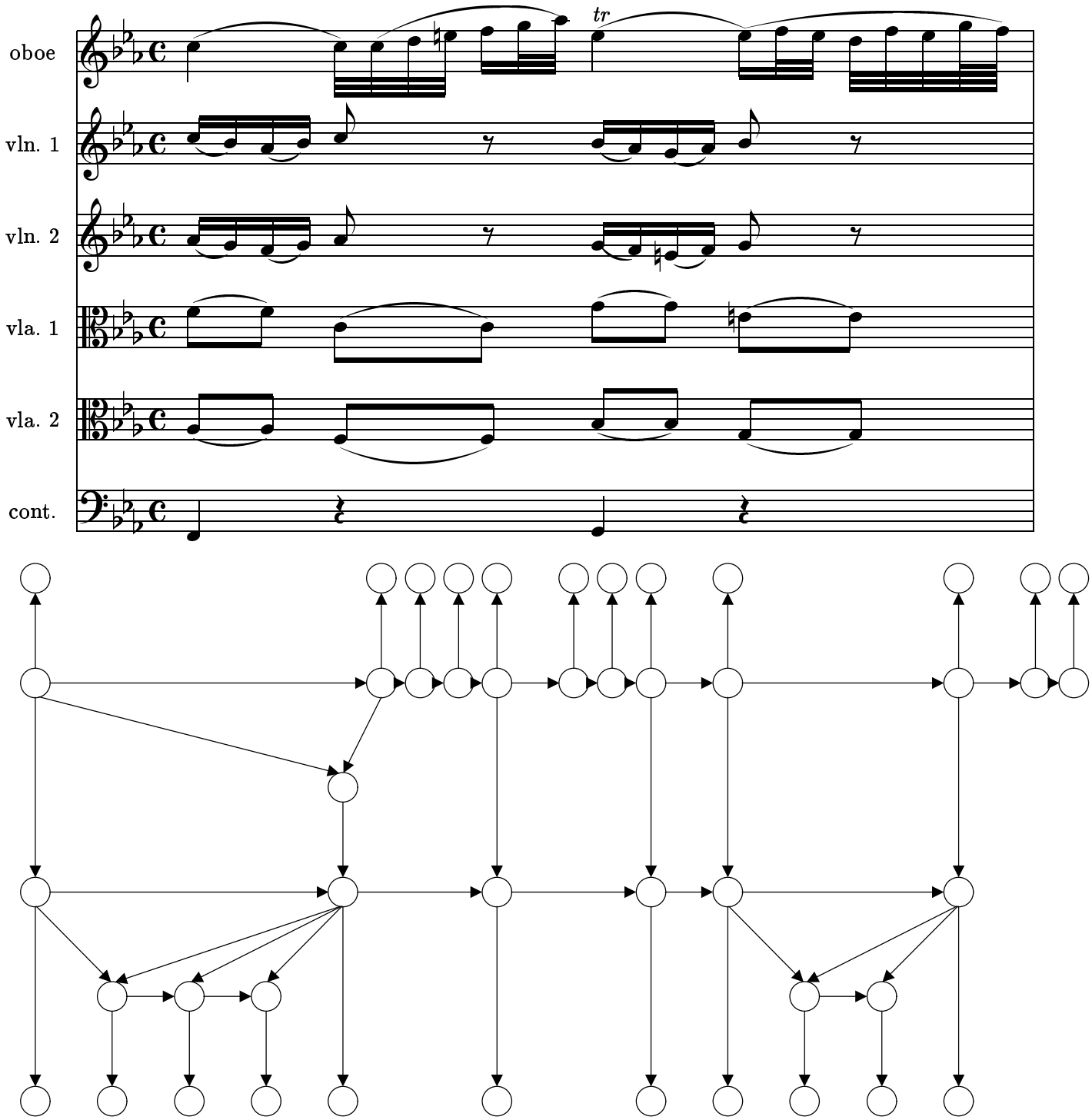

Figure 4: *Top:* The opening measures of the Sinfonia from J.S. Bach's Cantata 12. *Bottom:* The graph corresponding to the first 7/8 of the first measure for this music. The nodes in the 1st (top) layer correspond to the estimated solo note times that come from the Listen process ; the 2nd layer represents the solo process; the 3rd layer represents the phantom nodes (not discussed here); the 4th layer represents the coincident accompaniment nodes; the 5th layer represents the nodes that are "sandwiched" between coincident accompaniment nodes; the 6th (bottom) layer represents the actual accompaniment note times.

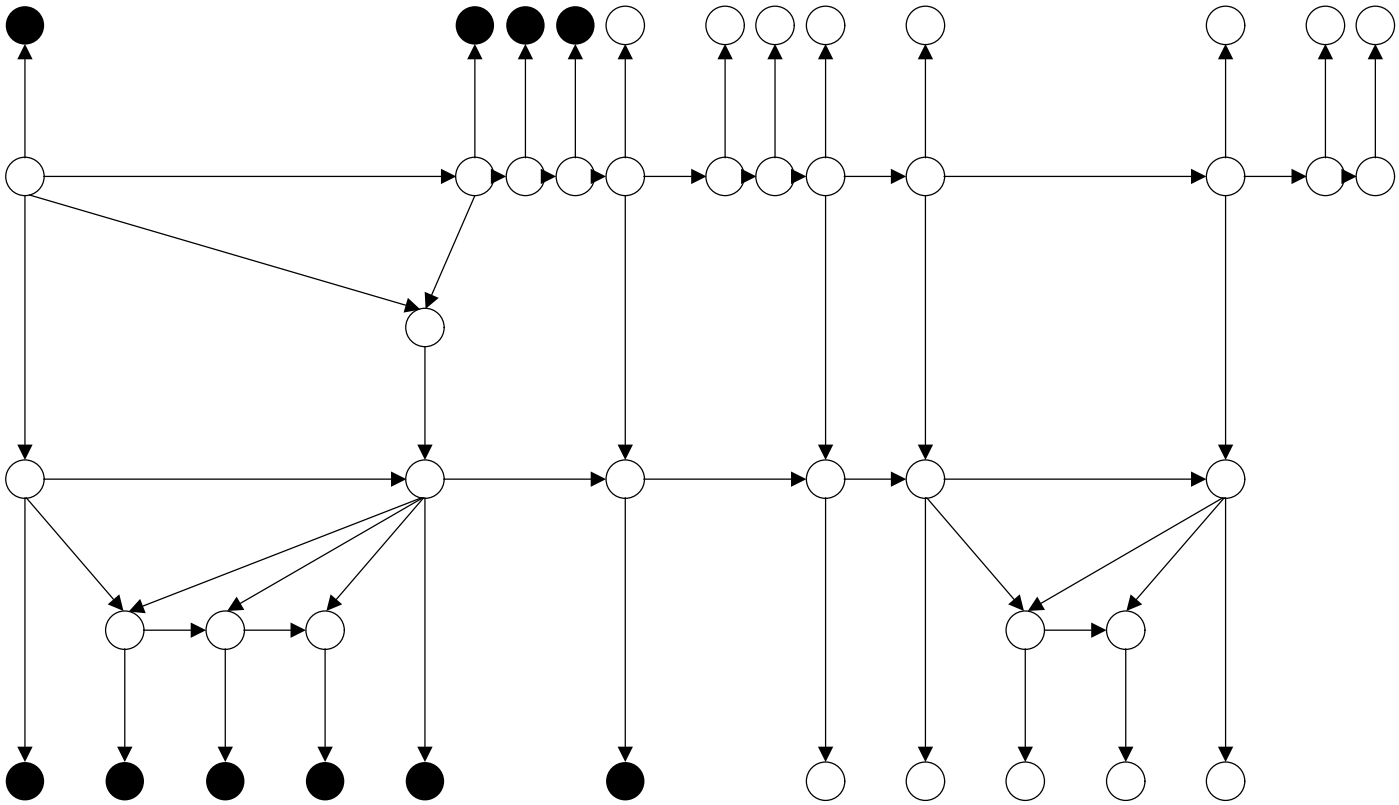

Figure 5: The graph corresponding to the opening notes of the Bach Cantata shown in Figure 4. The solid circles represent random vectors that have been observed at the current time, while the open circles represent currently unobserved random vectors.

simplify matters by focusing our training on a subset of these parameters as in Section 5, in which we consider only the training of the $\{\xi_n^{\text{solo}}\}$ parameters. To this end let $\xi_L = (\xi_{l_1}, \ldots, \xi_{l_P})$ be the independent vectors whose distributions we wish to learn — the "learnable" vector. Suppose that these distributions have parameters $\mu_1, \ldots, \mu_P$ and $\Sigma_1, \ldots, \Sigma_P$, known collectively as θ. Let x_O be the vector of variables that we can observe directly — the "observable" vector. For the situation discussed in Section 5, ξ_L will be composed of the vectors $x_0^{\text{solo}}, \xi_1^{\text{solo}}, \ldots, \xi_N^{\text{solo}}$ and x_O will be composed of the $x_0^{\text{obs}}, \ldots, x_N^{\text{obs}}$ vectors.

Our training is based on the following observation. Given an assignment of values to θ and a realization of the observable vector, $x_O = \alpha$, the posterior distribution on all model variables is computable via the message passing algorithm. If we regard the posterior mean of ξ_{l_p} as an estimate of ξ_{l_p}, and we had multiple such estimates from independent rehearsals, we could use them to reestimate the parameters θ. This is the essence of the EM algorithm. The actual updating scheme is as follows.

Let m_p^j and S_p be the posterior mean and variance of ξ_{l_p} given $x_O^j = \alpha^j$ (i.e. given the jth rehearsal) for $j = 1, \ldots, J$ and $p = 1, \ldots, P$; m_p^j and S_p will be computed as a byproduct of the message passing algorithm. The updating of the EM algorithm reduces in our case to

$$\mu_p^{\text{new}} = \frac{1}{J} \sum_{j=1}^{J} m_p^j$$

$$\Sigma_p^{\text{new}} = S_p + \frac{1}{J} \sum_{j=1}^{J} m_p^j m_p^{j^t} - \mu_p^{\text{new}} \mu_p^{\text{new}^t}$$

for $p = 1, \ldots, P$. This updating scheme is iterated and is guaranteed to converge to a local maximum of the data likelihood function.

5 Demonstration

We demonstrate here experiments performed with the Sinfonia from J.S. Bach's Cantata 12: "Weinen, Klagen, Sorgen, Zagen" ("Weeping, Crying, Sorrowing, Sighing") whose opening bar is shown in Figure 4 (Top). The movement is scored for solo oboe, violins, violas, and continuo, although we have transcribed it for oboe and "piano." The latter is, of course, not a real piano, but rather the output of an Alesis QSR tone generator driven through the standard MIDI (Musical Instrument Digital Interface) protocol. In this, and all of our examples, we prefer to score our accompaniment for percussion instruments, such as the piano, or plucked string instruments, such as the harp. For such instruments, the evolution of a note is largely deterministic between its inception and its end (either by damping or natural decay). Perhaps for this reason, these instruments are easier to synthesize and their artificial counterparts sound more like the real instruments. But also, we prefer to work with instruments whose expressive power can be harnessed by controlling only the onset times, initial "velocities" and end times of each note, since these are the only parameters we currently use in the accompaniment's performance.

We have developed a simple "mark-up" language for representing the musical score. In this language we represent the pitches and musical lengths of all notes, both solo and accompaniment, as well as any other information we wish to explicitly include in the performance. In our current example, the accompaniment dynamics are deterministically set using our mark-up language using terraced dynamics, crescendi, and diminuendi. Thus, at present, the dynamics are constant over different performances.

The accompaniment's conditional distribution given the solo part is also entered directly. In the present example, we have set the parameters of the conditional distribution to parsimoniously control the accompaniment's rhythmic flexibility. We arrived at our eventual settings through trial and error and have no reason to suppose that these parameters are set optimally. Although we have not done so yet, we propose to learn from examples the distribution governing both rhythmic nuance and dynamics for the accompaniment part in future work.

Most onset times of the solo process can be fairly accurately predicted from past history, however, others cannot. For example, it is virtually impossible to estimate when the first solo note will begin if there is no introduction from the accompaniment; nor could one anticipate the time the soloist will resume after a fermata or hold. Musicians deal with these situations by giving visual cues to one another, however such information is not available to our program since we only use audio input. To address this problem, we divide the score up into a sequence of "phrases." The playing of each of the phrases progresses exactly as described in Section 3. However, when we come to the end of a phrase, the next phrase is not begun until the first solo note in the new phrase has been detected. If this solo note coincides with an accompaniment event, then the accompaniment event will always be late since the solo note cannot be detected until after it has begun. This error might or might not be musically significant, depending on the magnitude of the error and the musical context, but we feel it is unavoidable without some other input from the soloist. From the standpoint of our probabilistic modeling, the phrases are regarded as independent random vectors. Thus there is no "connection" between adjacent solo notes that overlap a phrase boundary and the initial solo state of the phrase is an independent variable, as is x_0^{solo} in our model. In the current example, we divided the score into four phrases.

The solo parameters, $\{\mu_n^{\text{solo}}\}$ and $\{\Sigma_n^{\text{solo}}\}$, for each of these phrases are learned during a rehearsal phase. During this phase the player chooses a section of music, typically a single phrase, and performs the solo part to this section along with the accompaniment as played by the program. The player's acoustic signal is then reanalyzed off-line to estimate the times of the solo events more accurately and this information is saved. We then run the training algorithm using this, and all previous rehearsals, to update the solo parameters and the process is iterated. The rehearsal phrase then progresses much as it does between humans with the soloist setting an example and the accompanist learning what must be done to accommodate that example while maintaining a sense of internal musicality. The first several rehearsals are usually a little rough as the soloist must struggle to be true to his or her own musical ideals while not being distracted by a somewhat quirky accompaniment. But the accompaniment settles in quickly and begins to produce some musically satisfying results after only a few iterations.

Measuring the performance of an accompaniment system such as ours presents difficulties due to the many different criteria one might employ. The accompanist has primarily two simultaneous objectives: Playing musically and synchronizing with the soloist. While measuring success in the first objective is subjective, we can quantify the accuracy in synchronization. Figure 6 shows the empirical distribution of these differences in synchronization, measured in milliseconds, for the 203 points of coincidence in a performance of our system on Debussy's *Reverie*. While our system makes an occasional error greater than 100 milliseconds, the figure shows that such errors are uncommon.

Needless to say, no amount of discussion can answer the most important question: "How does it sound?" We rehearsed the Bach Sinfonia with the program for what amounts to about 15 complete renditions. A performance based on these rehearsals can be heard from the web page: http://fafner.math.umass.edu/music_plus_one/expert.htm The interested reader is encouraged to listen to this and other examples of our work in this area, including the Debussy example,

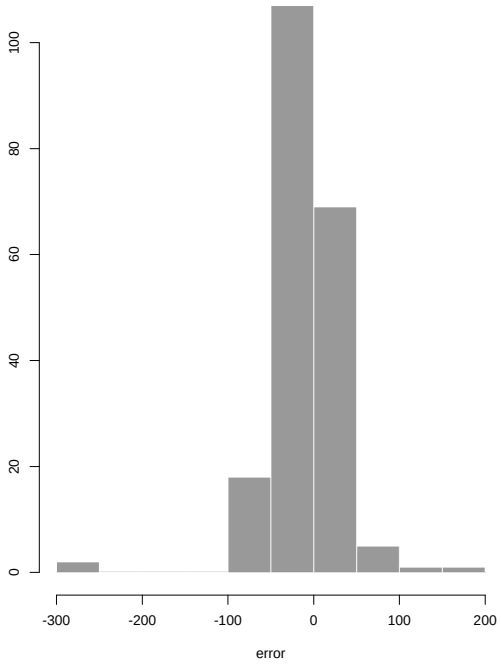

Figure 6: A histogram of the error, in milliseconds, between the times of solo notes and accompaniment notes that occur at identical score positions. Negative values indicate the accompaniment is early; positive values indicate the accompaniment is late.

referenced on the web page.

References

[Dannenberg 84] R. Dannenberg , "An On-Line Algorithm for Real-Time Accompaniment," *Proceedings of the International Computer Music Conference, 1984* IRCAM Paris, France, 193–198, 1984.

[Bloch & Dannenberg 85] J. Bloch, R. Dannenberg, "Real-Time Computer Accompaniment of Keyboard Performances," *Proceedings of the International Computer Music Conference, 1985* 279–289, Burnaby, British Columbia, Canada, 1985.

[Dannenberg & Mukaino 88] R. Dannenberg, H. Mukaino "New Techniques for Enhanced Quality of Computer Accompaniment" *Proceedings of the International Computer Music Conference, 1988* 243–249, Köln, 1988.

[Vercoe & Puckette 85] B. Vercoe, M. Puckette "Synthetic Rehearsal: Training the Synthetic Performer," *Proceedings of the International Computer Music Conference, 1985* 275–278, Burnaby, British Columbia, Canada, 1985.

[Vercoe 84] B. Vercoe, "The Synthetic Performer in the Context of Live Performance," *Proceedings of the International Computer Music Conference, 1984* 199–200, IRCAM Paris, France, 1984.

[Raphael 99] Raphael C. (1999), "Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 21, No. 4, pp. 360–370.

[Lauritzen 96] Lauritzen S. L., (1996), "Graphical Models," Oxford University Press, New York.

[Spiegelhalter et. al. 93] Spiegelhalter D., Dawid A. P., Lauritzen S., Cowell R. (1993), "Bayesian Analysis in Expert Systems," *Statistical Science,* Vol. 8, No. 3, pp. 219–283.

[Jensen 96] Jensen F., (1996), "An Introduction to Bayesian Networks," Springer-Verlag, New York.

[Lauritzen 92] Lauritzen S. L. (1992), "Propagation of Probabilities, Means, and Variances in Mixed Graphical Association Models," *Journal of the American Statistical Association*, Vol. 87, No. 420, (Theory and Methods), pp. 1098–1108.

[Lauritzen 95] Lauritzen S. L. (1995), "The EM Algorithm for Graphical Association Models with Missing Data," *Computational Statistics and Data Analysis*, Vol. 19, pp. 191–201.