

# A Probabilistic Expert System for Automatic Musical Accompaniment\*

Christopher Raphael†

March 22, 1999

## Abstract

A methodology is presented that allows a computer to play the role of musical accompanist in a non-improvised musical composition for soloist and accompaniment. The modeling of the accompaniment incorporates a number of distinct knowledge sources including timing information extracted in real-time from the soloist's acoustic signal, an understanding of the soloist's interpretation learned from rehearsals, and prior knowledge that guides the accompaniment toward musically plausible renditions. The solo and accompaniment parts are represented collectively as a large number of Gaussian random variables with a specified conditional independence structure — a Bayesian Belief Network. Within this framework a principled and computationally feasible method for generating real-time accompaniment is presented that incorporates the relevant knowledge sources. The EM algorithm is used to adapt the accompaniment to the soloist's interpretation through a series of rehearsals. A demonstration is provided from J.S. Bach's Cantata 12.

Keywords: Bayesian belief networks, conditional Gaussian distribution, EM algorithm, graphical model, music.

## 1 Introduction

The work presented here was inspired by a truly ingenious, if flawed, product known as “Music Minus One” (MMO). The typical MMO scenario begins with a composition for solo instrument and accompaniment, such as a violin sonata or piano concerto. All necessary players are assembled and two performances are recorded; the first, performed in the usual way, merely serves as an example of the way the solo part and accompaniment interact; in the second recording, however, the soloist is removed from the studio and the remaining players record only the accompaniment. This second, *accompaniment-only* recording is the focal point of MMO. A player of the featured instrument would, instrument in hand, play the recording on the stereo and try to add the missing part with all appropriate musical expression.

In practice, what follows is a battle of interpretive wills invariably resulting in the unconditional surrender of the live player. All of the soloist's musical nuances are, of course, ignored by the

---

\*This work was supported by a grant from the Five College Multimedia Access Project

†Department of Mathematics and Statistics, University of Massachusetts at Amherst, Amherst, MA 01003-4515, [raphael@math.umass.edu](mailto:raphael@math.umass.edu)

recording which can only respond with dogged determinism. Contrary to both musical etiquette and common sense, the soloist must follow the accompaniment.

Although MMO is often a desirable alternative to no accompaniment at all, it is the antithesis of what the music-making experience should ideally be: The accompaniment should follow the soloist and not vice-versa. This weakness motivates our answer to MMO which we call “Music Plus One” (MPO). Our goal is to create an expert system that “listens” to a soloist and creates a real-time non-improvised accompaniment that responds flexibly and musically to the soloist’s nuances of phrasing. As the piece is “rehearsed” the program *learns* the soloist’s interpretation and assimilates it into its accompaniment in future renditions. Thus Music Plus One will *add* to the live musician’s enjoyment by providing a tireless and sensitive accompanist.

Such a program would be an invaluable learning aid for instrumentalists and vocalists alike, providing a more realistic and satisfying musical context than isolated practice can. It would be useful as a teaching device in preparing for performance, for learning new music, and acquainting students with the demands of ensemble playing. And it would make music more accessible to the amateur player.

Several other researchers have addressed this automatic accompaniment problem, most notably Dannenberg et. al. [1],[2],[3] and Vercoe et. al. [4], [5] have had substantial success. In fact, Dannenberg’s research has even spawned a commercially available program. There are two significant distinctions between our work and that of Dannenberg and Vercoe, as follows.

First of all, our approach is model-based: We define a probabilistic model that describes the time evolution of a single part as well as the interaction between soloist and accompaniment. Our model serves as a link between our application and a wealth of useful mathematical knowledge. In particular, algorithms for optimal prediction and unsupervised learning follow naturally from our modeling assumptions. In addition, our model serves as a unifying framework allowing us to represent the many different knowledge sources the accompanist must utilize in a common language. Thus, our model allows the accompaniment to make a principled decision when it is faced with conflicting musical objectives.

Secondly, our approach is driven by the belief that learning is fundamental to satisfying musical performance. This is why musicians rehearse. Thus we have developed a framework that allows us to learn many different aspects of musical performance within a single learning methodology. Our experiments here focus on learning the soloist’s rhythmic interpretation, however, the same ideas could be used to learn the accompaniment’s rhythmic and dynamic interpretation, as well as the learning the accuracy of our acoustic “sensor.” Vercoe’s work also learns some aspects of the soloist’s interpretation while Dannenberg’s does not.

While we do not share any specific methodology or application, there is a strong philosophical connection between our work and that of Beran and Mazzola [13] and Kashino and Hagita [18]; in both cases, statistical methodology brings power and structure that has not been exploited elsewhere in applications to music, to our knowledge. Other examples of attempts at some form of computer-generated musical accompaniment are Dannenberg and Mont-Reynaud [16] which deals with accompanying an improvised solo line; Bryson [14], [15], which addresses chordal analysis; and Baird et. al. [19] which generates a real time accompaniment using MIDI input.

Our system is composed of a chain of hardware components that conceptually form a loop. The actual input to the system comes from a microphone focussed on the live player. When the program decides, in response to what it has “heard,” that it is time to play an accompaniment note, a 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. In this sense we figuratively “close the loop” in the chain of components.

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 [6] 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. Most often the detection of a note occurs shortly after the true onset of the note, however some events such as rearticulations and notes following trills can best be detected with the benefit of more hindsight, so there is variable latency in their detection. 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.

## 1.1 The Problem: Knowledge Fusion

Producing a musically satisfying accompaniment requires the integration of a number of disparate sources of information which we describe here. In fact, a central challenge of our work is representing these knowledge sources (KSs) in a common framework that is computationally tractable.

**KS 1:** In standard Western musical notation, most note lengths are represented in terms of some idealized unit such as beats or measures. One model for the actual note duration suggests dividing the idealized note length by the tempo; for example, a quarter note at 120 quarter notes per minute lasts .5 seconds. Typically the tempo is not constant, but rather varies gradually throughout the evolution of the piece. The actual note lengths prescribed by the score coupled with the expectation of a slowly varying tempo provide the first knowledge source in our system.

**KS 2:** Our system aspires to accompany a live player in real time, so it must also have real-time information describing the soloist’s performance. This information is provided by the Listen process as a sequence of messages passed to the Play process, one for each solo note. The messages contain the estimated times at which the solo notes begin, and are delivered to the Play process with variable latency.

**KS 3:** Musicians regard rehearsal as prerequisite to good performance. It is during rehearsal that musicians establish habits and develop expectations that allow them to function as a group. We believe that the knowledge gleaned from rehearsal is even more important to our system since its interpretation of the real-time information will always be less sophisticated than the human’s. Thus we require that our system learns the tempo and rhythmic nuances of the soloist and uses them in subsequent performances.

**KS 4:** For a performance to be aesthetically satisfying, in addition to synchronizing with the soloist, the accompanist must play with an internal musicality. For instance, tempo changes should be gradual unless otherwise indicated by some musical consideration. Our experience suggests that it is not possible to learn all necessary musicality through imitation of the soloist. Rather, the accompanist needs some prior information that will guide it toward a musically satisfying accompaniment.

In addition to respecting all of the above knowledge sources, our system must satisfy an additional constraint. 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.

In this paper we construct a probabilistic model that combines all of the above knowledge sources. In particular, Section 2 presents a joint probability distribution on the solo and accompaniment parts which models both tendencies in the individual parts as well as interactions

between them. Our whole orientation toward the automatic accompaniment problem is guided by the formidable real-time computational demand; Section 3 describes a principled and feasible real-time algorithm for generating the accompaniment. Section 4 shows how the rhythmic interpretation of the soloist can be learned automatically through rehearsal, thereby informing future performances. Finally, Section 5 demonstrates our approach on the Sinfonia from J.S. Bach’s Cantata 12. Such a demonstration is, of course, incomplete without an audio example; since it is not possible to include such an example directly into our paper, we have made one available through a web page given in Section 5.

## 2 The 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 greatly 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 computational 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

Consider Fig. 1 which shows the time evolution for several renditions of the opening of Robert Schumann’s *3rd Romance for Oboe and Piano*. We see from this example that there is considerable variation in the tempo as exhibited by the non-linear shape of the plots. In addition, we see that a good deal of the tempo variation is repeatable from performance to performance, suggesting that an accompanist might benefit from “learning” the soloist’s interpretation. In this section 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  $l_0, \dots, l_N$ ; 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  $(t_n, s_n)^t$ ,  $n = 0, \dots, N$ , represent the “state” of the  $n^{\text{th}}$  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,

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

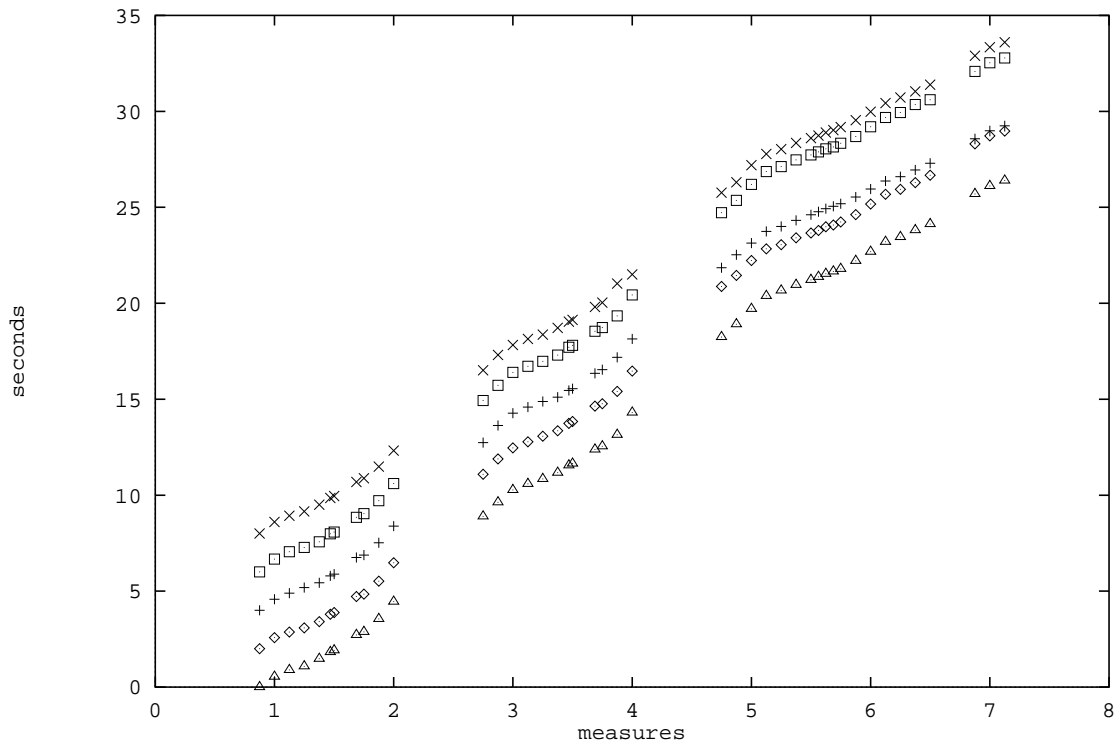


Figure 1: Musical time (measures) vs. real time (secs) for five different performances of the first 7 measures of Robert Schumann's 3rd Romance for Oboe and Piano. Note that the tempo fluctuations are significant, yet highly repeatable.

for  $n = 1, \dots, N$  where  $\tau_1, \dots, \tau_{n-1}$  and  $\sigma_1, \dots, \sigma_{n-1}$  are random variables. If the  $\{(\tau_n, \sigma_n)^t\}$  vectors are removed from Eqn. 1, then the tempo process is constant,  $s_0 = s_1 = \dots = s_N$ , 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 introduces two kinds of flexibility into our model. A positive value of  $\sigma_n$  means that the measure size is increasing at the  $n^{\text{th}}$  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  $\tau_n$  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  $\tau_n$  would compress a note. Musicians might debate whether it is possible to have a lengthening or shortening of times between note onsets without a corresponding change in tempo. At the very least the inclusion of the  $\{\tau_n\}$  terms allow the model to explain note durations that differ from those prescribed by the current tempo without actually changing the tempo. This adds stability to the model when it is used for prediction.

Our model is expressed more compactly and more completely as

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

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

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

$$x_n^{\text{solo}} = \begin{pmatrix} t_n \\ s_n \end{pmatrix} \quad (4)$$

$$\xi_n^{\text{solo}} = \begin{pmatrix} \tau_n \\ \sigma_n \end{pmatrix} \quad (5)$$

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

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

and

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

$n = 1, \dots, N$ .

## 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  $x_0^{\text{obs}}, \dots, x_N^{\text{obs}}$ , where  $x_n^{\text{obs}}$  is the estimated onset time for the  $n^{\text{th}}$  solo note. How do these times relate to the idealized state process  $x_0^{\text{solo}}, \dots, x_{n-1}^{\text{solo}}$ ? Our model assumes that

$$x_n^{\text{obs}} = A_n^{\text{obs}} x_n^{\text{solo}} + \xi_n^{\text{obs}} \quad (8)$$

where

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

and

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

with  $\xi_0^{\text{obs}}, \dots, \xi_N^{\text{obs}}$  mutually independent and independent of  $x_0^{\text{solo}}$  and  $\xi_1^{\text{solo}}, \dots, \xi_N^{\text{solo}}$  as well. Thus, the  $x_0^{\text{obs}}, \dots, x_N^{\text{obs}}$  are “noisy” observations of the idealized note onset times  $t_0, \dots, t_N$ . There are at least two factors that contribute to the discrepancy between  $t_n$  and  $x_n^{\text{obs}}$ . 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 or 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.

The model for  $x_0^{\text{solo}}, \dots, x_N^{\text{solo}}$  and  $x_0^{\text{obs}}, \dots, x_N^{\text{obs}}$  presented so far is an example of the Kalman Filter. The rich theory from Kalman Filters can be used to answer several musically relevant questions one might want to ask, as follows.

Suppose we know  $\mu_0^{\text{solo}}, \dots, \mu_N^{\text{solo}}, \Sigma_0^{\text{solo}}, \dots, \Sigma_N^{\text{solo}}$ , and  $\sigma_0^{\text{obs}}, \dots, \sigma_N^{\text{obs}}$ . Suppose the partial output from our Listen process for the first several solo note times is  $x_0^{\text{obs}}, \dots, x_n^{\text{obs}}$  and we wish to use these to estimate the idealized time and tempo,  $x_n^{\text{solo}}$ , for the  $n^{\text{th}}$  note. The Kalman Filter gives a simple recursion for computing the conditional distribution of  $x_n^{\text{solo}}$  given  $x_0^{\text{obs}}, \dots, x_n^{\text{obs}}$ . Thus a natural method for providing a running tempo and position estimate results easily. Furthermore, the distribution of *future* note times can be computed by repeatedly applying Eqn. 2 to the conditional distribution of  $x_n^{\text{solo}}$ . Such predictions would prove useful for guiding an accompaniment.

Another musically relevant question concerns the training of our model from past performances. The distributions of the  $\{\xi_n^{\text{solo}}\}$  variables provide a characterization of the rhythmic variation of the solo distribution. In particular, the means of these random vectors describe places of tempo variation, elongation or compression of notes, while the covariances give information about the repeatability of these tendencies. Collectively, these parameters characterize the rhythmic interpretation of the soloist; estimating these parameters will be the way in which we learn from rehearsal. Given a complete (and noisy) observation of all solo notes,  $x_0^{\text{obs}}, \dots, x_N^{\text{obs}}$ , the Kalman Smoother can be used to compute the posterior distribution on each  $x_n^{\text{solo}}$ . This idea can be extended easily to compute posterior distributions for the  $\{\xi_n^{\text{solo}}\}$  vectors. Many learning algorithms follow naturally from this point and we will discuss one based on the EM algorithm in Section 4.

The theory of the Kalman Filter would actually be sufficient for our purposes were these the only questions of interest. Unfortunately, the theory of the Kalman Filter is restricted to the very specific dependency structure developed so far, and does not generalize, in any obvious way, to a model incorporating both solo *and* accompaniment parts. It is fortunate, however, that the theory of Bayesian Belief Networks (BBNs) extends the domain covered by the Kalman Filter. This framework will form the theoretical backbone of our approach.

## 2.3 Representing Probabilities on Graphs

Let  $G = (\Gamma, \Lambda)$  be a directed acyclic graph (DAG) where  $\Gamma$  is a collection of graph nodes and  $\Lambda$  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  $x_\Delta$  for the random vector composed of components of  $x$  associated with members of  $\Delta$ . By convention, we write  $x_\gamma$  instead of the more correct  $x_{\{\gamma\}}$  and we write  $x_\Gamma$  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

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

where the parents of  $x_{\gamma}$ ,  $\text{pa}(x_{\gamma})$ , are nodes in  $\Gamma$  that have edges leading to  $\gamma$  and  $f_{\gamma}$  is the conditional density for  $x_{\gamma}$  given its parents  $x_{\text{pa}(\gamma)}$  [7]. If  $A, B, S \subset \Gamma$ , we use the notation  $x_A \perp x_B | x_S$  to mean that  $x_A$  and  $x_B$  are conditionally independent given  $x_S$ . Eqn. 9 can be shown [7] under fairly general conditions to be equivalent to the *directed local Markov property*,

$$x_{\gamma} \perp x_{\beta} | x_{\text{pa}(\gamma)} \quad (10)$$

where  $\beta \in \text{nd}(\gamma)$ , the non-descendants of  $\gamma$ . When Eqn. 9 holds, one can sample from the density  $f$  by making a sequence of independent choices: First choose independently the vectors  $\{x_{\gamma}\}$  where  $\text{pa}(\gamma) = \emptyset$  according to their marginal distributions,  $f_{\gamma}(\cdot | \emptyset)$ . Then for each remaining  $x_{\gamma}$ , once its parents have been chosen, choose  $x_{\gamma}$  from its conditional distribution given its parents. Thus, Eqn. 9 suggests a generative model for the probability distribution.

If  $y = (y_1, y_2)^t$  is a partition of a Gaussian random vector with mean

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$$

and covariance

$$\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$

then the conditional distribution of  $y_1$  given  $y_2$  is Gaussian with mean and covariance

$$\begin{aligned} \mu_{y_1|y_2} &= \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(y_2 - \mu_2) \\ \Sigma_{y_1|y_2} &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \end{aligned}$$

provided the stated inverse exists. Thus, given  $y_2$ , one could sample from the conditional distribution for  $y_1$  by letting

$$y_1 = Ay_2 + \xi$$

where  $A = \Sigma_{12}\Sigma_{22}^{-1}$  and  $\xi \sim N(\mu_1 - \Sigma_{12}\Sigma_{22}^{-1}\mu_2, \Sigma_{y_1|y_2})$ . Then if  $f$  from Eqn. 9 is the density for a Gaussian distribution, we can sample from the density by recursive application of

$$x_{\gamma} = A_{\gamma}x_{\text{pa}(\gamma)} + \xi_{\gamma} = \sum_{\alpha \in \text{pa}(\gamma)} A_{\alpha \rightarrow \gamma}x_{\alpha} + \xi_{\gamma} \quad (11)$$

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  $f_{\gamma}(x_{\gamma} | x_{\text{pa}(\gamma)})$ . In light of this discussion, we see that a Gaussian distribution that admits recursive factorization with respect to a given DAG can be specified simply by choosing the  $\{A_{\gamma}\}$ ,  $\{\mu_{\gamma}\}$ , and  $\{\Sigma_{\gamma}\}$  components. This is the representation we will employ.

As an example, the joint distribution of  $\{x_0^{\text{solo}}, \dots, x_N^{\text{solo}}\}$  and  $\{x_0^{\text{obs}}, \dots, x_N^{\text{obs}}\}$ , was defined through throughout Eqns. 2 and 8, both of the same form as Eqn. 11. Thus the joint distribution of these random variables respects the graph of Fig. 2 with the indicated association between random variables and nodes. In this graph every node, except the node corresponding to  $x_0^{\text{solo}}$  has a single ‘‘parent’’ node. For each of the nodes in Figure 2, the assumptions of the previous sections specify the  $\{A_{\gamma}\}$ ,  $\{\mu_{\gamma}\}$ , and  $\{\Sigma_{\gamma}\}$  components.

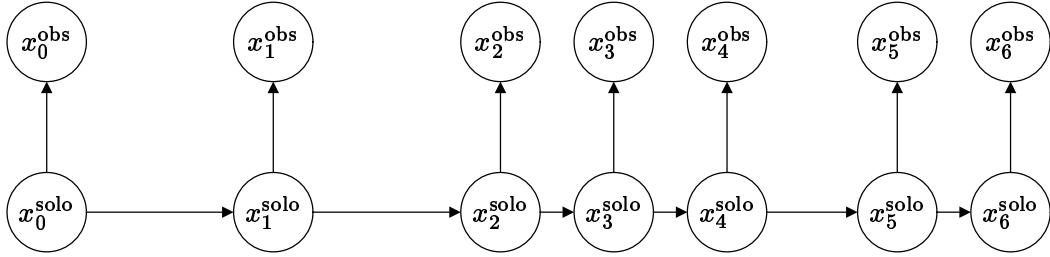


Figure 2: A graphical representation of the joint distribution on  $x_0^{\text{solo}} \dots, x_N^{\text{solo}}$  and  $x_0^{\text{obs}} \dots, x_N^{\text{obs}}$ . The horizontal placement of nodes is proportional to their onset times in *musical* time (measures). We follow this convention in all subsequent graphs.

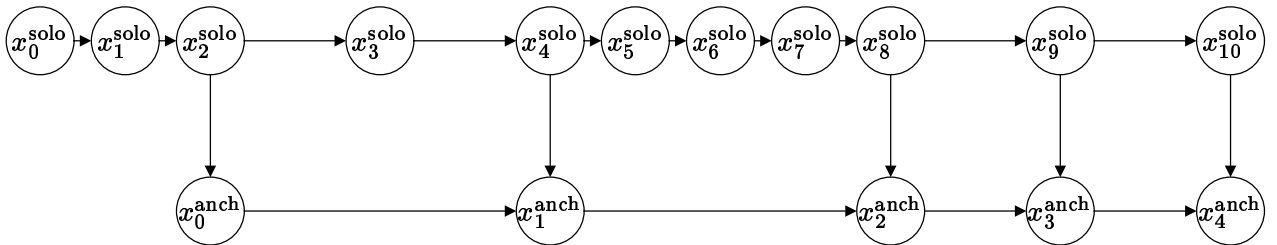
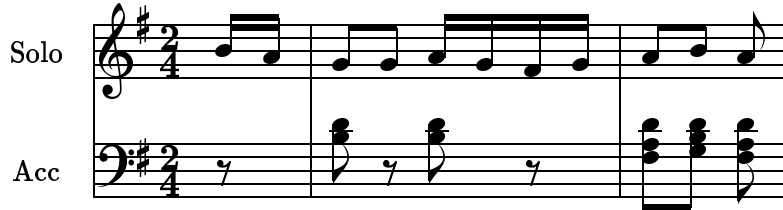


Figure 3: *Top*: A musical example in which each accompaniment event is coincident with a solo event. *Bottom*: A graphical representation of the conditional dependency structure.

## 2.4 Adding the Accompaniment

In this section we provide an overview of our modeling of the interplay between solo and accompaniment by discussing two generic musical scenarios. These two examples generalize to cover most of the situations encountered in practice. Section 2.5 contains a more thorough presentation of our accompaniment model by explicitly defining the conditional distribution of the accompaniment variables given the solo variables.

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 3. 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 and internal consistency are occasionally in opposition to one another. Our goal is to model a compromise between these two competing objectives.

Let  $x_0^{\text{solo}}, \dots, x_N^{\text{solo}}$  and  $x_0^{\text{anch}}, \dots, x_M^{\text{anch}}$  be the solo state process and accompaniment state process. The accompaniment random vectors,  $\{x_m^{\text{anch}}\}$ , will be used to “anchor” other accompaniment variables not yet introduced, hence their name. 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 3. Suppose  $x_n^{\text{solo}}$  and  $x_m^{\text{anch}}$  are coincident solo and accompaniment notes. From the graph we see that, given  $x_n^{\text{solo}}$ ,  $x_{n+1}^{\text{solo}}$  is independent of  $x_k^{\text{anch}}$  for  $k \leq m$ . In contrast, given  $x_m^{\text{anch}}$ ,  $x_{m+1}^{\text{anch}}$  is, in general, not independent of  $x_k^{\text{solo}}$  for  $k \leq n$ . 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 4 in which there are several accompaniment events between a pair of accompaniment events that coincide with the solo notes marked  $A$  and  $B$ . The center of Figure 4 shows the most obvious extension of our modeling to this situation. Consider the random vectors labeled  $x_A, x_B, x_C$ , corresponding to the events labeled  $A, B, C$  in the musical fragment. By Eqn. 10 we see that  $x_B \perp x_C | x_A$ . This, however, is undesirable since, for instance, an expansion in the time between notes  $A$  and  $B$  should result in a proportional expansion in the time between  $A$  and  $C$ . We instead model this situation as in the bottom panel of Fig. 4 in which we make explicit the dependence of  $x_C$  on *both*  $x_A$  and  $x_B$ .

The above examples form show the basic ideas we use in modeling the conditional distribution of the accompaniment, given the solo part. The following section specifies this conditional distribution more completely. Much of this next section is summarized in Figure 5 and the details of this section can be skipped on first reading once the essence of this figure is grasped.

## 2.5 Explicit Modeling of the Accompaniment

We begin with a probabilistic model of the accompaniment alone, which we refer to as the “practice room” distribution for the accompaniment. This model embodies the behavior of the accompaniment when the soloist is not present, so only issues of musicality and internal consistency are important. From the viewpoint of our model, the accompaniment consists of a sequence of  $M + 1$  events, or times at which one or more accompaniment notes begin or end. Our practice room model for the accompaniment process,  $y_0, \dots, y_M$ , is perfectly analogous to the solo model of Eqns. 2-7:

$$y_m = B_{m-1}y_{m-1} + \eta_m \tag{12}$$

for  $m = 1, \dots, M$  where

$$y_m = \begin{pmatrix} t'_m \\ s'_m \end{pmatrix}$$

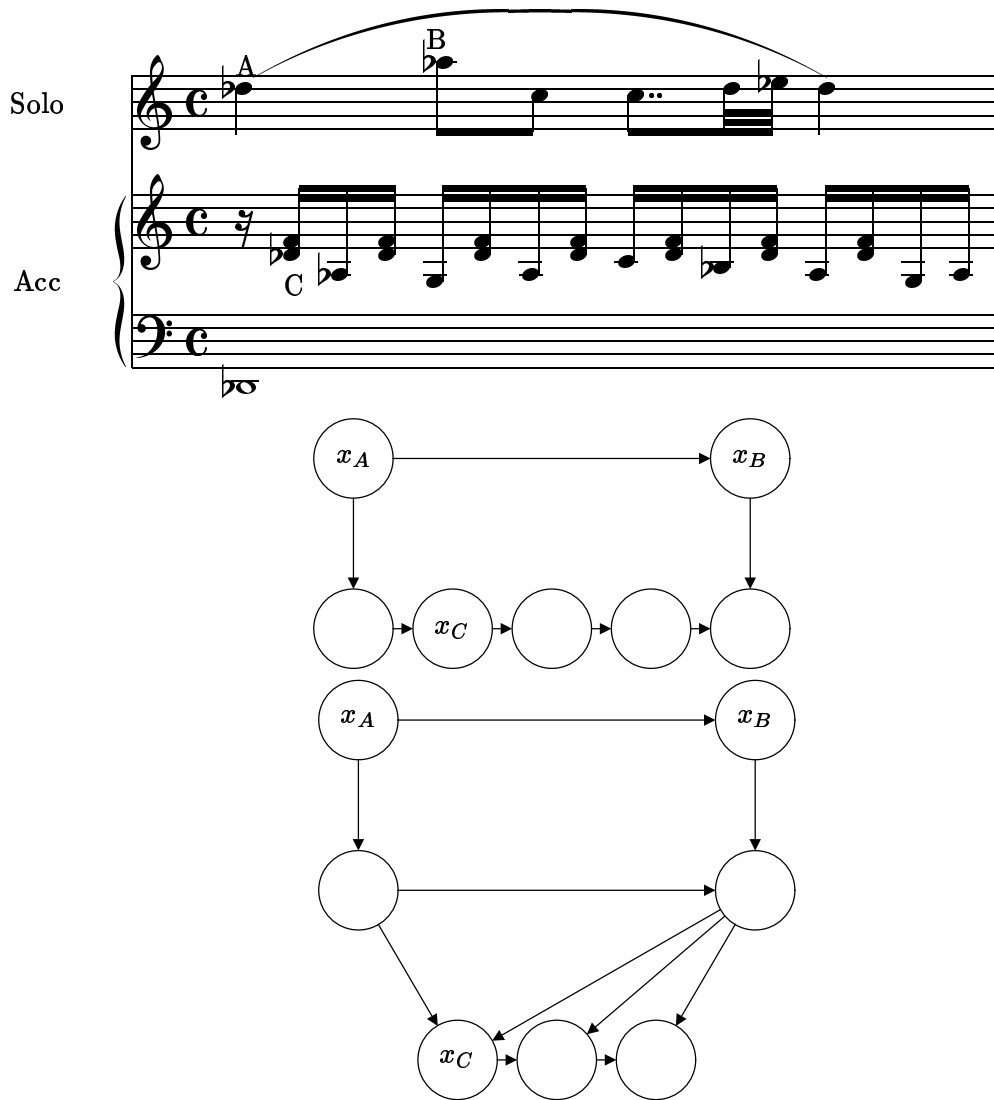


Figure 4: *Top*: A musical excerpt beginning with four accompaniment events evenly spaced over the first solo note. *Center*: One possible graphical representation of dependency structure. *Bottom*: Another possible graphical representation.

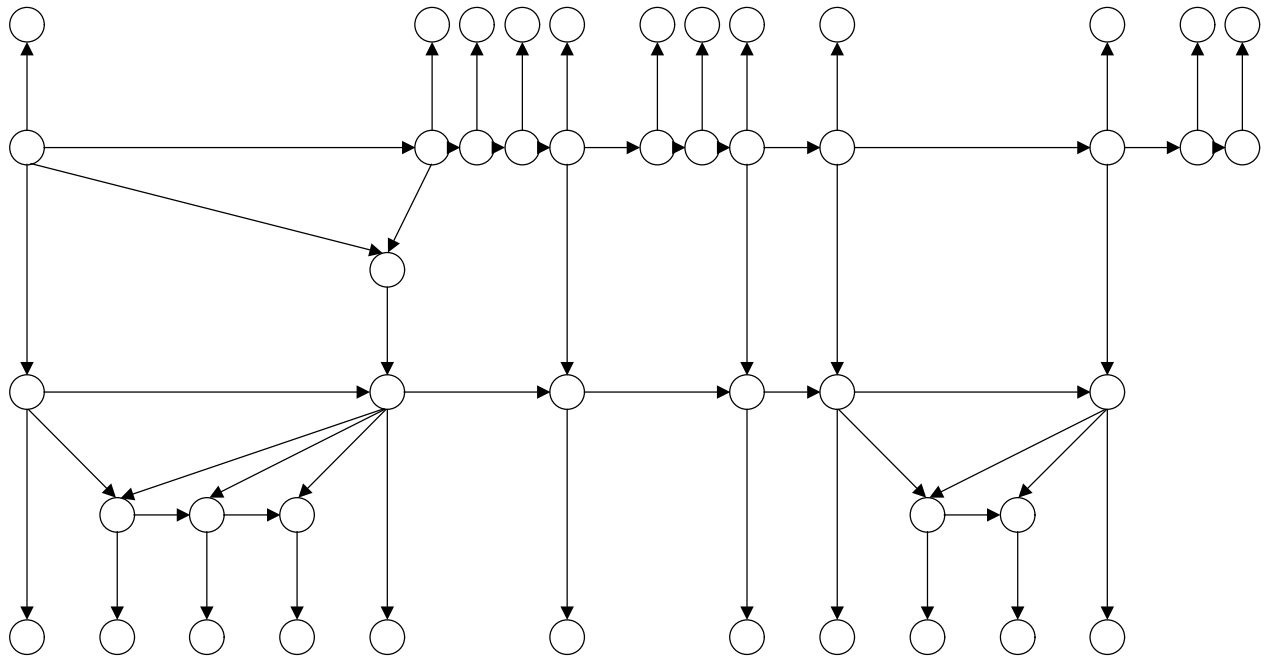


Figure 5: *Top*: The opening measure of the Sinfonia from J.S. Bach's Cantata 12. *Bottom*: The graph corresponding to the first 7/8 of the first measure for this music. The nodes in the 1st (top) layer correspond to the estimated solo note times that come from the Listen process and are modeled by Eqn. 8; the 2nd layer represents the solo process of Eqn. 2; the 3rd layer represents the phantom nodes of Eqn. 13; the 4th layer represents the anchor nodes of Eqn. 14; the 5th layer represents the sandwiched nodes of Eqn. 18; the 6th (bottom) layer represents the actual accompaniment note times of Eqn. 19.

where  $t'_m$  and  $s'_m$  are the onset time, in seconds, and the tempo, in seconds per measure, for the  $m^{\text{th}}$  accompaniment event;

$$B_m = \begin{pmatrix} 1 & l'_m \\ 0 & 1 \end{pmatrix}$$

with  $l'_m$  the length, in measures, of the  $m^{\text{th}}$  accompaniment event; and  $\eta_1, \dots, \eta_M$  are random 2-vectors such that  $y_0$  and  $\eta_1, \dots, \eta_M$  are mutually independent random vectors with

$$\begin{aligned} y_0 &\sim N(\mu'_0, \Sigma'_0) \\ \eta_m &\sim N(\mu'_m, \Sigma'_m) \end{aligned}$$

$m = 1, \dots, M$ .

Sections 2.1 and 2.2 describe the joint distribution on the  $x_0^{\text{solo}}, \dots, x_N^{\text{solo}}$  and  $x_0^{\text{obs}}, \dots, x_N^{\text{obs}}$  variables. These variables correspond to the top two layers in the graph of Fig. 5 which describes the dependency structure on *all* variables in our model. The following three subsections describe the conditional distribution on all other model variables, given the variables of the top two layers.

### 2.5.1 Modeling Phantom Variables

Let  $p^{\text{solo}}(n) = \sum_{k=0}^{n-1} l_k$  be the onset time, in measures, of the  $n^{\text{th}}$  solo event and let  $p^{\text{acc}}(m) = \sum_{k=0}^{m-1} l'_k$  be the onset time of the  $m^{\text{th}}$  accompaniment event. We define the functions

$$\begin{aligned} l(m) &= \max\{n : 0 \leq n \leq N, p^{\text{solo}}(n) \leq p^{\text{acc}}(m)\} \\ r(m) &= \min\{n : 0 \leq n \leq N, p^{\text{solo}}(n) \geq p^{\text{acc}}(m)\} \end{aligned}$$

for  $m = 0, \dots, M$  representing the indices of the closest left and right solo event neighbors of the  $m^{\text{th}}$  accompaniment event. For simplicity we assume here that both  $p^{\text{solo}}(0) = p^{\text{acc}}(0)$  and  $p^{\text{solo}}(N) = p^{\text{acc}}(M)$  so that  $l(m)$  and  $r(m)$  are always defined, however this requirement can be dropped with some minor modifications. We define the *sandwiched* note indices,  $S$ , to be

$$S = \{m : 0 \leq m \leq M, l(m) = l(m+1), r(m) = r(m-1)\}$$

Thus  $S$  indexes accompaniment events whose left and right accompaniment neighbors are sandwiched between the same pair of adjacent solo events. For instance, the 2nd, 3rd, and 4th, accompaniment events in the music of Fig. 5 are sandwiched events since all have the property that their left and right accompaniment neighbors lie between the 1st and 2nd solo notes. The remaining event indices are said to be *non-sandwiched*.

If a non-sandwiched accompaniment event,  $m$ , is not coincident with any solo event, ( $l(m) \neq r(m)$ ), then we define a *phantom* vector,  $x_m^{\text{phan}}$ , whose parents are the vectors  $x_{l(m)}^{\text{solo}}$  and  $x_{r(m)}^{\text{solo}}$ .  $x_m^{\text{phan}}$  contains the idealized time and tempo for the  $m^{\text{th}}$  accompaniment event when only synchronicity is considered. In particular we define

$$x_m^{\text{phan}} = A_{m,l}^{\text{phan}} x_{l(m)}^{\text{solo}} + A_{m,r}^{\text{phan}} x_{r(m)}^{\text{solo}} + \zeta_m^{\text{phan}} \quad (13)$$

where

$$\begin{aligned} A_{m,r}^{\text{phan}} &= \begin{pmatrix} \lambda_m & 0 \\ 0 & \lambda_m \end{pmatrix} \\ A_{m,l}^{\text{phan}} &= \begin{pmatrix} 1 - \lambda_m & 0 \\ 0 & 1 - \lambda_m \end{pmatrix} \end{aligned}$$

where

$$\lambda_m = \frac{p^{\text{acc}}(m) - p^{\text{solo}}(l(m))}{p^{\text{solo}}(r(m)) - p^{\text{solo}}(l(m))}$$

and

$$\xi_n^{\text{phan}} \sim N(0, \Sigma^{\text{phan}})$$

for some fixed covariance  $\Sigma^{\text{phan}}$ . See the third layer of Fig. 5. In essence, a phantom event is an imaginary random vector that describes the time and tempo that the solo part would have if it had played an event coincident with the  $m$ th accompaniment event. The time and tempo of the phantom event are modeled as convex combinations of the times and tempi of its left and right solo neighbors. For example, a phantom event which, according to the score, lies halfway between two solo events would be a random vector whose mean time and tempo are halfway between the times and tempi of the solo neighbors.

### 2.5.2 Modeling Anchor Vectors

For each  $m \in S^c$ , we now define an *anchor* vector,  $x_m^{\text{anch}}$ . For  $m \in S^c$  define the upper parent to be

$$x_m^{\text{up}} = \begin{cases} x_{l(m)}^{\text{solo}} & l(m) = r(m) \text{ (i.e. if a coincident solo event exists)} \\ x_m^{\text{phan}} & \text{otherwise} \end{cases}$$

and for each  $m \in S^c \setminus 0$  let the left parent be

$$x_m^{\text{left}} = x_{a(m)}^{\text{anch}}$$

where

$$a(m) = \max\{0 \leq k < m : k \in S^c\}$$

We define  $x_0^{\text{up}}$  to be the only parent of  $x_0^{\text{anch}}$  and model the conditional distribution of  $x_0^{\text{anch}}$  given  $x_0^{\text{up}}$  by

$$x_0^{\text{anch}} = x_0^{\text{up}} + \xi_0^{\text{anch}}$$

where  $\xi_0^{\text{anch}} \sim N(0, \Sigma^{\text{up}})$  for some fixed covariance matrix  $\Sigma^{\text{up}}$ . For the remaining  $m \in S^c$ ,  $x_m^{\text{anch}}$  has two parents:  $x_m^{\text{left}}$  and  $x_m^{\text{up}}$ . In defining the conditional distribution of  $x_m^{\text{anch}}$  given  $x_m^{\text{up}}$  and  $x_m^{\text{left}}$  we seek a compromise between two competing objectives: faithfulness to the solo part and adhering to the practice room distribution of the accompaniment, as follows.

As mentioned before, we want the accompaniment to follow the soloist and the “solo opinion” of the distribution of  $x_m^{\text{anch}}$  could be modeled as

$$\text{M1: } x_m^{\text{anch}} \sim N(x_m^{\text{up}}, \Sigma^{\text{up}})$$

On the other hand, the “accompaniment opinion” of the distribution of  $x_m^{\text{anch}}$  could be obtained from the practice room distribution in the following way. By the model of Eqn. 12, the conditional distribution of  $y_l$ , given  $y_{a(m)}$ , for  $a(m) < l < m$ , can be expressed as

$$y_l | y_{a(m)} \sim N(R_{m,l} y_{a(m)} + d_{m,l}, \Delta_{m,l})$$

where

$$\begin{aligned} R_{m,a(m)} &= I \\ d_{m,a(m)} &= 0 \\ \Delta_{m,a(m)} &= 0 \end{aligned}$$

and

$$\begin{aligned} R_{m,l} &= B_{l-1} R_{m,l-1} \\ d_{m,l} &= B_{l-1} d_{m,l-1} + \mu'_l \\ \Delta_{m,l} &= B_{l-1} \Delta_{m,l-1} B_{l-1}^t + \Sigma'_l \end{aligned}$$

for  $l = a(m) + 1 \dots, m$ .

Equating  $y_m$  with  $x_m^{\text{anch}}$  and  $y_{a(m)}$  with  $x_{a(m)}^{\text{anch}}$  we find an alternative model for  $x_m^{\text{anch}}$ ,

$$\text{M2} : x_m^{\text{anch}} \sim N(R_{m,m} x_{a(m)}^{\text{anch}} + d_{m,m}, \Delta_{m,m})$$

We seek a compromise between these two different opinions. Thus, we model the conditional distribution of  $x_m^{\text{anch}}$  given  $x_m^{\text{up}}$  and  $x_m^{\text{left}}$  by taking the density proportional to the product of the two density functions given in M1 and M2. Then, by completing the square,

$$x_m^{\text{anch}} | x_m^{\text{left}}, x_m^{\text{up}} \sim N(\Sigma_m^{\text{anch}} (\Sigma_m^{\text{up}^{-1}} x_m^{\text{up}} + \Delta_{m,m}^{-1} (R_{m,m} x_{a(m)}^{\text{anch}} + d_{m,m})), \Sigma_m^{\text{anch}})$$

where  $\Sigma_m^{\text{anch}} = (\Sigma_m^{\text{up}^{-1}} + \Delta_{m,m}^{-1})^{-1}$ . Thus

$$x_m^{\text{anch}} = A_{a(m)}^{\text{anch}} x_{a(m)}^{\text{anch}} + A_m^{\text{up}} x_m^{\text{up}} + \xi_m^{\text{anch}} \quad (14)$$

where

$$\begin{aligned} \xi_m^{\text{anch}} &\sim N(\Sigma_m^{\text{anch}} \Delta_{m,m}^{-1} d_{m,m}, \Sigma_m^{\text{anch}}) \\ A_{a(m)}^{\text{anch}} &= \Sigma_m^{\text{anch}} \Delta_{m,m}^{-1} R_{m,m} \\ A_m^{\text{up}} &= \Sigma_m^{\text{anch}} \Sigma_m^{\text{up}^{-1}} \end{aligned}$$

The anchor vectors are illustrated pictorially in the 4th layer of Figure 5.

### 2.5.3 Modeling Sandwiched Notes

We now model the accompaniment events associated with the *sandwiched* indices,  $S$ . Let  $a(m)$  and  $m$  be two adjacent anchor indices, so  $a(m)+1, \dots, m-1 \subset S$ . We model the accompaniment events associated with these indices by random vectors  $x_{a(m)+1}^{\text{sand}}, \dots, x_{m-1}^{\text{sand}}$ . The conditional distribution for these vectors is derived completely from the practice room distribution. In particular, we take the conditional distribution of the  $x_{a(m)+1}^{\text{sand}}, \dots, x_{m-1}^{\text{sand}}$  given  $x_{a(m)}^{\text{anch}}, x_m^{\text{anch}}$  to be the same as the conditional distribution of the  $y_{a(m)+1}, \dots, y_{m-1}$  given  $y_{a(m)}, y_m$ .

To accomplish this we begin by computing the distribution of  $y_{a(m)+1} | y_{a(m)}, y_m$ . Then by the Markov structure governing the  $\{y_l\}$  process we see that, for  $a(m) + 1 < l < m$ , the conditional distribution of  $y_l$  given  $y_{a(m)}, \dots, y_{l-1}, y_m$  depends only  $y_{l-1}, y_m$ . This dependency structure is indicated in the 5th layer in Figure 5. The conditional distributions are computed as follows.

If we view the conditional density of  $y_m$  given  $y_l$  as a function of  $y_l$ , then we can represent this function as

$$f_{y_m | y_l}(y_m | y_l) \propto \mathcal{N}(y_l ; T_{m,l} y_m + c_{m,l}, \Phi_{m,l}) \quad (15)$$

where  $\mathcal{N}(\cdot ; \mu, \Sigma)$  is the joint normal density function with mean vector  $\mu$  and covariance matrix  $\Sigma$ . It can be shown that the following recursion holds for the parameters  $\{T_{m,l}, c_{m,l}, \Phi_{m,l}\}$ ,

$$\begin{aligned} T_{m,m} &= I \\ c_{m,m} &= 0 \\ \Phi_{m,m} &= 0 \end{aligned}$$

and

$$\begin{aligned} T_{m,l} &= B_l^{-1} T_{m,l+1} \\ c_{m,l} &= B_l^{-1} c_{m,l+1} - \mu'_{l+1} \\ \Phi_{m,l} &= B_l^{-1} \Phi_{m,l+1} B_l^{-1t} + \Sigma'_{l+1} \end{aligned}$$

for  $l = a(m) + 1, \dots, m - 1$ . Note also that Eqn. 12 gives the conditional density of  $y_l$  given  $y_{l-1}$ ,

$$f_{y_l|y_{l-1}}(y_l|y_{l-1}) = \mathcal{N}(y_l; B_{l-1}y_{l-1} + \mu'_l, \Sigma'_l) \quad (16)$$

Then when both  $f_{y_m|y_l}$  and  $f_{y_l|y_{l-1}}$  are viewed as functions of  $y_l$ , we have

$$\begin{aligned} f_{y_m|y_l}(y_m|y_l) f_{y_l|y_{l-1}}(y_l|y_{l-1}) &= f_{y_m, y_l|y_{l-1}}(y_m, y_l|y_{l-1}) \\ &\propto f_{y_l|y_m, y_{l-1}}(y_l|y_m, y_{l-1}) \end{aligned}$$

so by completing the square,

$$y_l|y_{l-1}, y_m \sim N(\Sigma_l^{\text{sand}}(\Sigma_l'^{-1}(B_{l-1}y_{l-1} + \mu'_l) + \Phi_{m,l}^{-1}(T_{m,l}y_m + c_{m,l})), \Sigma_l^{\text{sand}}) \quad (17)$$

where  $\Sigma_l^{\text{sand}} = (\Sigma_l'^{-1} + \Phi_{m,l}^{-1})^{-1}$ . Thus, equating  $x_{a(m)+1}^{\text{sand}}, \dots, x_{m-1}^{\text{sand}}$  with  $y_{a(m)+1}, \dots, y_{m-1}$  in Eqn. 17 we have

$$x_l^{\text{sand}} = A_{l,l-1}^{\text{sand}} x_{l-1}^{\text{sand}} + A_{m,l}^{\text{sand}} x_m^{\text{anch}} + \xi_l^{\text{sand}} \quad (18)$$

for  $l = a(m) + 1, \dots, m - 1$  where we define  $x_{a(m)}^{\text{sand}} = x_{a(m)}^{\text{anch}}$  where

$$\begin{aligned} A_{l,l-1}^{\text{sand}} &= \Sigma_l^{\text{sand}} \Sigma_l'^{-1} B_{l-1} \\ A_{m,l}^{\text{sand}} &= \Sigma_l^{\text{sand}} \Phi_{m,l}^{-1} T_{m,l} \\ \xi_l^{\text{sand}} &\sim N(\Sigma_l^{\text{sand}} \Sigma_l'^{-1} \mu'_l + \Sigma_l^{\text{sand}} \Phi_{m,l}^{-1} c_{m,l}, \Sigma_l^{\text{sand}}) \end{aligned}$$

The sandwiched events are illustrated in the 5th layer of Figure 5.

Finally, we create one last layer of random variables. Associated with each accompaniment event we create an observable variable  $x_m^{\text{out}}$  corresponding to the time at which the event is played,

$$x_m^{\text{out}} = \begin{cases} D x_m^{\text{sand}} & \text{if } m \in S \\ D x_m^{\text{anch}} & \text{otherwise} \end{cases} \quad (19)$$

where  $D = (1, 0)$ . These ‘‘output’’ random variables compose the lowest layer in the graph of Figure 5.

### 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:  $x_0^{\text{obs}}, \dots, x_{n(t)}^{\text{obs}}$  and the accompaniment events already played:  $x_0^{\text{out}}, \dots, x_{m(t)}^{\text{out}}$ . Figure 6 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

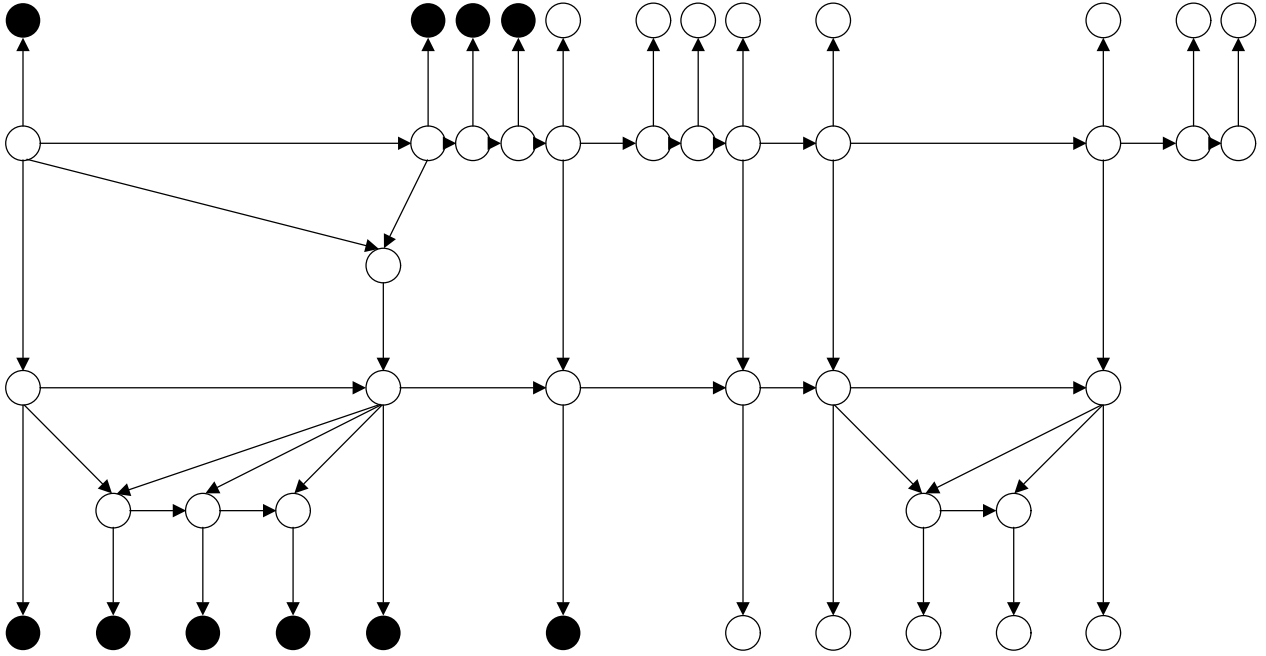


Figure 6: The graph corresponding to the opening notes of the Bach Cantata shown in Figure 5. The solid circles represent random vectors that have been observed at the current time, while the open circles represent currently unobserved random vectors.

the knowledge sources discussed in Section 1.1: 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. Our current system, discussed in Section 5, schedules  $x_{m(t)+1}^{\text{out}}$  to sound at the conditional mean of the distribution, but one could argue for more exotic choices justified by the dissimilar consequences of scheduling events early and late. 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 [8], [7], [9], 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 [10]. A brief account of the “evidence propagation” algorithm is given here, but we assume the reader is familiar with the basic ideas.

It is well known that, if a probability is specified on a DAG, marginal distributions can be computed by operating on a triangulation of the associated *undirected* graph. To form this graph one simply “moralizes” the graph by marrying all unmarried parents — connecting all pairs of nodes that have a common child — then all directed edges are replaced by undirected edges and edges are added until all cycles of length greater than three are cut by a chord. We then find the cliques of the triangulated graph and connect them into a graph of cliques known as a “clique

tree.” The triangulation process guarantees that the clique graph will in fact be a tree. The clique tree will have the following property making information passing possible: For any two cliques,  $C_1$  and  $C_2$ , any variables in their intersection must be contained in every clique along the unique path between  $C_1$  and  $C_2$  in the clique tree.

If we let  $\mathcal{C}$  denote the set of cliques and  $\mathcal{S}$  be the *separators* — intersections between neighboring cliques in the clique tree, then the joint probability density can be represented by

$$\phi(x_\Gamma) = \frac{\prod_{C \in \mathcal{C}} \phi_C(x_C)}{\prod_{S \in \mathcal{S}} \phi_S(x_S)} \quad (20)$$

where the  $\{\phi_S(x_S)\}$  and  $\{\phi_C(x_C)\}$  are “potential functions” depending only on the indicated variables. To create one such representation, note that every factor in Eqn. 9 is a function of variables contained in at least one of the cliques. Each  $\phi_C(x_C)$  for  $C \in \mathcal{C}$  is then taken to be the product of all associated factors from Eqn. 9; the  $\phi_S$  for  $S \in \mathcal{S}$  are initialized to 1. If we disregard the multiplicative constant, each factor in Eqn. 9 is the exponential of some quadratic form in the indicated variables, so our representation of factors in Eqn. 20 consists only of the parameters of the quadratic form.

Through a series of “flow” operations, one can transform Eqn. 20 into an “equilibrium” representation of the joint density in which the  $\phi_C$  and  $\phi_S$  functions of Eqn. 20 are the marginal densities of the associated variables. From such a representation one can essentially “read off” the marginal distribution on any particular variable. The flow operation from clique  $C_1$  to a neighboring clique  $C_2$  modifies only the potential  $\phi_{C_2}$  and the separating potential  $\phi_S$  by

$$\begin{aligned} \phi_S^{\text{new}} &= \sum_{C_1 \setminus S} \phi_{C_1} \\ \phi_{C_2}^{\text{new}} &= \phi_{C_2}^{\text{old}} \frac{\phi_S^{\text{new}}}{\phi_S^{\text{old}}} \end{aligned}$$

where the  $\sum_A$  notation means integrating out all variables in  $A$ . It is not difficult to compute the quadratic form parameters of  $\phi_S^{\text{new}}$  and  $\phi_{C_2}^{\text{new}}$  as functions of the parameters for  $\phi_S^{\text{old}}, \phi_{C_2}^{\text{old}}, \phi_{C_1}$  as in [10].

To find the equilibrium, every clique must both receive and send flows to its neighboring cliques. There are several different flow schedules that lead to the equilibrium; we have used the so-called “Hugin” propagation scheme [9]. In this scheme some clique is arbitrarily chosen to be the root and then two recursive routines, *Collect Evidence* and *Distribute Evidence*, are performed. In *Collect Evidence* the root requests that each of its neighbors send a flow into the root which recursively make similar requests of their neighbors (other than the requesting clique). If a clique has no such neighbors, then it will send the requested flow immediately; otherwise, the flow is sent from a clique once that clique has received all incoming flows. When this sequence of message passing ends  $\phi_{C_{\text{root}}}$  is equal to the marginal density for the variables in  $C_{\text{root}}$ . The equilibrium representation is then achieved by performing *Distribute Evidence* which simply reverses the sequence of message passing. Starting with the root, each clique now sends a flow to each neighbor. Once a clique has received all incoming flows it will send flows to each of its neighbors (other than the clique it received from).

Returning to our application, when new information is received in the form of a played accompaniment event or a detected solo event, we can update our representation to describe the posterior distribution on all model variables conditioned on the new information. The new information can be represented as  $x_v = c$  for some variable  $x_v$ . We then multiply the potential of the clique containing  $v$  by the function  $1_{\{x_v=c\}}$ , after which our potential representation is proportional to the

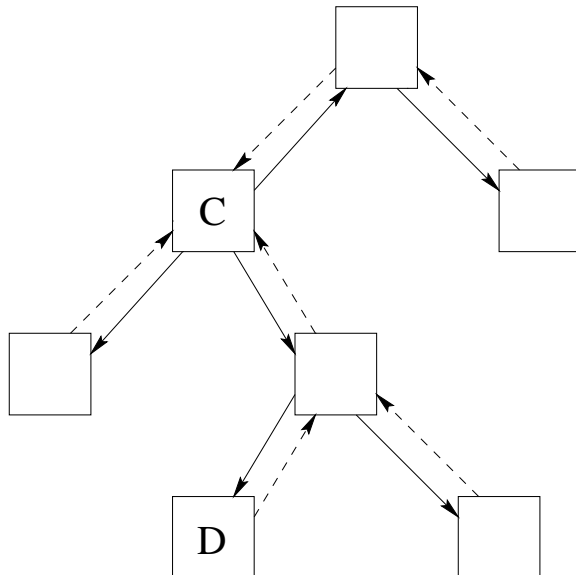


Figure 7: In the above clique tree, solid arrows denote pending flows while dashed arrows denote non-pending flows. After a variable in  $C$  is observed, all flows moving away from  $C$  are marked as pending. To compute the marginal distribution on clique  $D$ ,  $D$  need only receive all pending flows.

posterior distribution. After incorporating information in this way, the potential representation is no longer in equilibrium, however the equilibrium can be reestablished by a round of the message passing algorithm.

In principle, the playing of the accompaniment could proceed by performing both phases of message passing every time a new piece of information is received, however, this would place an unreasonable and unnecessary computational burden on our real-time playing algorithm. Rather, we proceed as follows (see Figure 7). We begin with a potential representation that is in equilibrium (this computation can be performed off-line). When a variable is observed we update the clique,  $C$ , containing the variable as above. We next mark each flow moving away from  $C$  as pending, thus half of the possible flows in the clique tree will be marked. We then update the distribution on the clique containing the next unplayed accompaniment event which lies in some clique  $D$ . To accomplish this, we traverse the clique tree in the order of *Collect Evidence* treating  $D$  as the root and only performing the flows that are marked as pending. On completion of this abbreviated version of *Collect Evidence*,  $\phi_D$  will be proportional to the marginal distribution of the variables in  $D$  conditioned on all currently observed variables. At this point the tree is not in equilibrium, but we maintain our record of all currently pending flows for future calculations.

## 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. The basic idea we use for training our BBN is thoroughly developed for the case of discrete variables in [11], and is briefly sketched for the joint Gaussian case which applies to our situation.

The probabilistic model for the entire collection of random variables,  $x_\Gamma$ , 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  $x_\Gamma$ . For instance, the distributions of the  $\{\xi_n^{\text{solo}}\}$  vectors model the soloist’s rhythmic interpretation; the distributions of the  $\{\xi_m^{\text{anch}}\}$ ,  $\{\xi_m^{\text{sand}}\}$ ,  $\{\xi_m^{\text{phan}}\}$  vectors describe the conditional distribution on the accompaniment variables given the solo variables; and the distributions of the  $\{\xi_n^{\text{obs}}\}$  model the precision of the Listen process. In principle, our training procedure can be employed to estimate any or all of the parameters governing the  $\{\xi_\gamma\}$  vectors. In practice, we 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  $\xi_L = (\xi_{l_1}, \dots, \xi_{l_P})$  be the independent vectors whose distributions we wish to learn — the “learnable” vector. Suppose that these distributions have parameters  $\mu_1, \dots, \mu_P$  and  $\Sigma_1, \dots, \Sigma_P$ , known collectively as  $\theta$ . Let  $x_O$  be the vector of variables that we can observe directly — the “observable” vector. For the situation discussed in Section 5,  $\xi_L$  will be composed of the vectors  $x_0^{\text{solo}}, \xi_1^{\text{solo}}, \dots, \xi_N^{\text{solo}}$  and  $x_O$  will be composed of the  $x_0^{\text{obs}}, \dots, x_N^{\text{obs}}$  vectors.

Our training is based on the following observation. Given an assignment of values to  $\theta$  and a realization of the observable vector,  $x_O = \alpha$ , the posterior distribution on all model variables is computable via the message passing algorithm of Section 3. In particular, the message passing algorithm gives a posterior marginal distribution on the variables in each clique. Each of the  $\{\xi_{l_p}\}$  vectors can be computed as a linear transformation of the variables in a clique, so we can compute the posterior distribution on each  $\xi_{l_p}$  vector from the equilibrium representation. If we regard the posterior mean as an estimate of  $\xi_{l_p}$ , and we had multiple such estimates from independent rehearsals, we could use them to reestimate the parameters  $\theta$ . This is the essence of the EM algorithm. A more detailed account follows.

Our training will be based on a series of  $J$  rehearsals. Thus there are now  $J$  instances of the random vector  $x_\Gamma$  denoted  $x_\Gamma^1, \dots, x_\Gamma^J$ , which we assume to be independent. We write  $\mathbf{x}_\Gamma = (x_\Gamma^1, \dots, x_\Gamma^J)$  and use similar notation for the observable vector  $\mathbf{x}_O = (x_O^1, \dots, x_O^J)$ , the learnable vector  $\xi_L = (\xi_L^1, \dots, \xi_L^J)$ , and the actual observation vector  $\alpha = (\alpha^1, \dots, \alpha^J)$ . Thus our observed information is summarized by  $\mathbf{x}_O = \alpha$ .

The well-known EM algorithm [12] starts with an initial value for the parameter,  $\theta^0$ , and produces a sequence of values  $\theta^1, \theta^2, \dots$  that is known to increase the likelihood at every iteration provided that a local maximum has not already been achieved and to converge to a local maximum of the likelihood function under mild assumptions. The EM iteration for the parameter of an exponential family computes  $\theta^{\text{new}} = \theta^{n+1}$  from  $\theta^{\text{old}} = \theta^n$  by solving

$$E_{\theta^{\text{new}}}[T(\mathbf{x}_\Gamma)] = E_{\theta^{\text{old}}}[T(\mathbf{x}_\Gamma)|\mathbf{x}_O = \alpha] \quad (21)$$

where  $T(\mathbf{x}_\Gamma)$  is the canonical sufficient statistic for  $\theta$ . Since  $\mathbf{x}_\Gamma$  is a random sample from a Gaussian distribution and our parameter  $\theta$  includes both means and covariances, this sufficient statistic is given by

$$T(\mathbf{x}_\Gamma) = \begin{pmatrix} \sum_{j=1}^J \xi_{l_1} & \cdots & \sum_{j=1}^J \xi_{l_P} \\ \sum_{j=1}^J \xi_{l_1}^j \xi_{l_1}^{j^t} & \cdots & \sum_{j=1}^J \xi_{l_P}^j \xi_{l_P}^{j^t} \end{pmatrix}$$

Let  $m_p^j$  and  $S_p$  be the posterior mean and variance of  $\xi_{l_p}^j$  given  $x_O^j = \alpha^j$  for  $j = 1, \dots, J$  and  $p = 1, \dots, P$ ;  $m_p^j$  and  $S_p$  will be computed as a byproduct of the message passing algorithm of the

previous section. Then from Eqn. 21 we have

$$E_{\theta^{\text{new}}} \left[ \sum_{j=1}^J \xi_{l_p}^j \right] = E_{\theta^{\text{old}}} \left[ \sum_{j=1}^J \xi_{l_p}^j \mid \mathbf{x}_O = \boldsymbol{\alpha} \right]$$

for  $p = 1, \dots, P$  so dividing both sides by  $J$  gives

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

Also from Eqn. 21 we have

$$E_{\theta^{\text{new}}} \left[ \sum_{j=1}^J \xi_{l_p}^j \xi_{l_p}^{j^t} \right] = E_{\theta^{\text{old}}} \left[ \sum_{j=1}^J \xi_{l_p}^j \xi_{l_p}^{j^t} \mid \mathbf{x}_O = \boldsymbol{\alpha} \right]$$

for  $p = 1, \dots, P$ . Since for a random vector  $\xi$  with mean  $\mu$  and covariance  $\Sigma$  we have  $E\xi\xi^t = \Sigma + \mu\mu^t$ , then

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

for  $p = 1, \dots, P$ . From this it follows that

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

.

## 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”). This Sinfonia, whose opening bar is shown in Figure 5 (Top), is a highly expressive movement filled with almost literal examples of the title words. Our choice of music challenges our program in a context that seems to require the musical sensitivity of a human musician.

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). Probably 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 (volumes) are deterministically set using our mark-up language using terraced dynamics, crescendos, and diminuendos. Thus, at present, the dynamics are constant over different performances.

The accompaniment’s “practice room” distribution is also entered directly into the score. In the present example, we have set the parameters of the practice room distribution to parsimoniously control the accompaniment’s rhythmic flexibility. For the accompaniment updates ( $\{\eta_m\}$  from Eqn. 12), all mean values for both tempo and position are set to 0, so all departures from metronomic performance are either in response to, or anticipation of, the evolution of the solo part. We input into the score different variances for the accompaniment tempo and position updates. A tempo or position update variable with relatively high variance is one that has more flexibility in its realization. In the absence of any other constraint, the variance of any update will be increasing in its corresponding note length, as one would expect to be the case. Aside from these general principles, we arrived at our eventual settings through trial and error and have no reason to suppose that these parameters are set optimally.

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  $x_0^{\text{solo}}$  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 rehearsals.

Table 1 quantifies what the accompaniment learned during the rehearsal phase on the Bach Sinfonia through an analysis of the predictive power of our model. In particular, Table 1 shows the distributions of  $k$ -note prediction errors defined by  $|x_n^{\text{obs}} - E[x_n^{\text{obs}} | x_0^{\text{obs}}, \dots, x_{n-k}^{\text{obs}}]|$  for  $k = 1, 2$ . These errors were computed on an example of the first phrase of the Bach Sinfonia that was not included in any of the training data. The two rows of the table corresponding to no training

training examples	.0-.025	.025-.05	.05-.10	.10-
0	25	23	30	13
3	32	24	33	2
10	59	24	5	3
20	60	22	5	4
0	13	9	31	37
3	33	21	28	8
10	53	20	13	4
20	48	24	13	5

Table 1: The top 4 rows of the table shows “one-note” prediction errors for our model when trained with different amounts of data. For example, the first row of the table means that with 0 training examples 25 of the notes were predicted with an error in the range of .0-.025 secs., 23 of the notes were predicted with an error in the range .025-.05 secs. etc. The bottom 4 rows display the same information for “two-note” prediction errors.

examples demonstrate that our model performs reasonably well, even when the model parameters are initialized to arbitrary values — in this case all  $\{\xi_n^{\text{solo}}\}$  random variables are initialized to have 0 mean and relatively unopinionated covariance matrices. This situation corresponds to the “sight-reading” task, as when two musicians play a piece together for the first time. Although our predictions do not have the benefit of training data at this point, they are still informed by the relative lengths of notes, as indicated in the score, and the expectation of gradually varying tempo which we have built directly into the model. Table 1 shows a marked decrease in predication errors after only 3 rehearsals, most obviously demonstrated by the .0-.025 column.

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](http://fafner.math.umass.edu/music_plus_one/expert.htm) The interested reader is encouraged to listen to this and other examples of our past work in this area referenced on the web page.

## 6 Future Work

Our current system is capable of representing a rich variety of musical interpretations in the accompaniment, however we currently must specify these interpretations by hand. Most notably, we must painstakingly set the conditional distribution of the accompaniment given the solo part, as well as all of the dynamics of the accompaniment. In fact, our willingness to separate the interplay of rhythm and dynamics into two noninteracting components comes from the difficulty of describing their interplay by hand. We believe the awkwardness of this process is an impediment to achieving musically satisfying results from the accompaniment in situations in which one cannot create a feeling of musicality simply by imitating the soloist. Our goal for the future is to learn this musicality through training data. In particular, we would like to learn a conditional distribution on the accompaniment part that involves interaction between both timing and dynamics since these two facets are intimately intertwined in human performance. We anticipate that the main challenge will be in learning the musicality that a human can demonstrate without also learning

the arbitrariness and inaccuracies that live players often succumb to. In short, we want to combine what the computer does well — accuracy and fidelity to the score, with what the human does well — expressiveness and musicality. Our proposed course of action will begin by creating low-dimensional parameterizations of “interpretation space” thus greatly limiting what will be learned from the live player. In addition to focusing the learning on areas that are known to be musically relevant, this helps to avoid the overfitting problem so often encountered when high dimensional parameters are learned from a limited supply of training data.

## References

- [1] R. Dannenberg, “An On-Line Algorithm for Real-Time Accompaniment,” *Proceedings of the International Computer Music Conference, 1984* IRCAM Paris, France, 193–198, 1984.
- [2] 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.
- [3] R. Dannenberg, H. Mukaino “New Techniques for Enhanced Quality of Computer Accompaniment” *Proceedings of the International Computer Music Conference, 1988* 243–249, Köln, 1988.
- [4] 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.
- [5] 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.
- [6] 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.
- [7] Lauritzen S. L., (1996), “Graphical Models,” Oxford University Press, New York.
- [8] Spiegelhalter D., Dawid A. P., Lauritzen S., Cowell R. (1993), “Bayesian Analysis in Expert Systems,” *Statistical Science*, Vol. 8, No. 3, pp. 219–283.
- [9] Jensen F., (1996), “An Introduction to Bayesian Networks,” Springer-Verlag, New York.
- [10] 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.
- [11] Lauritzen S. L. (1995), “The EM Algorithm for Graphical Association Models with Missing Data,” *Computational Statistics and Data Analysis*, Vol. 19, pp. 191–201.
- [12] Dempster A. P., Laird N. M., Rubin D. B. (1977) “Maximum Likelihood from Incomplete Data via the EM Algorithm,” *Journal of the Royal Statistical Society Series B*, Vol. 39, pp. 1–38.
- [13] Beran J., Mazzola G. (1999) “Analyzing Musical Structure and Performance — A Statistical Approach” *Statistical Science*, Vol. 14, No. 1, pp. 47–79.
- [14] Bryson J. (1992) “The Subsumption Strategy Development of a Music Modelling System,” *Masters’ Thesis*, Dept. of Artificial Intelligence, University of Edinburgh, 1992.

- [15] Bryson, J., (1995) "The Reactive Accompanist: Adaptation and Behaviour Decomposition in a Music System," in "The Biology and Technology of Intelligent Autonomous Agents," Ed. L. Steels, Springer-Verlag, London, 1995.
- [16] Dannenberg R., Mont-Reynaud B., (1987) "Following an Improvisation in Real Time," *Proceedings of the International Computer Music Conference, 1987* 241–258, 1984.
- [17] Camurri A., Capocaccia M., Zaccaria R., (1990) "ENA: Experimental Neural Accompanist," *International Neural Network Conference*, 1987.
- [18] Kashino K., Hagita N., (1996) "A Music Scene Analysis System with the MRF-Based Information Integration Scheme," *Proceedings of the International Conference on Pattern Recognition*, 1996 pp. 725–729.
- [19] Baird B., Blevins D., Zahler N., (1993) "Artificial Intelligence and Music: Implementing an Interactive Computer Performer," *Computer Music Journal*, vol. 17, no. 2, pp. 73–79.