Probabilistic Factor Oracles for Multidimensional Machine Improvisation

Ken Déguernel,* Emmanuel Vincent,* and Gérard Assayag†

*Multispeech Team
Inria
615 Rue du Jardin Botanique
54600 Villers-lès-Nancy, France
{ken.deguernel, emmanuel.vincent}@inria.fr
†Représentations Musicales
IRCAM
1 Place Igor Stravinsky
75004 Paris, France
{ken.deguernel, gerard.assayag}@ircam.fr

Abstract: This article presents two methods to generate automatic improvisation using training over multidimensional sequences. We consider musical features such as melody, harmony, timbre, etc., as dimensions. We first present a system combining interpolated probabilistic models with a factor oracle. The probabilistic models are trained on a corpus of musical work to learn the correlation between dimensions, and they are used to guide the navigation in the factor oracle to ensure a logical improvisation. Improvisations are therefore created in a way in which the intuition of a context is enriched with multidimensional knowledge.

We then introduce a system creating multidimensional improvisations based on communication between dimensions via probabilistic message passing. The communication infers some anticipatory behavior on each dimension influenced by the others, creating a consistent multidimensional improvisation.

Both systems were evaluated by professional improvisers during listening sessions. Overall, the systems received good feedback and showed encouraging results—first, on how multidimensional knowledge can improve navigation in the factor oracle and, second, on how communication through message passing can emulate the interactivity between dimensions or musicians.

Our goal is to design a system able to generate multidimensional musical improvisations. By "dimensions," we mean musical features such as melody, harmony, rhythm, timbre, etc. (cf. Bimbot et al. 2014). To achieve this goal, this system must be able to learn correlations between dimensions on a large musical corpus and, at the same time, be able to follow a local frame constructed from a musician's live playing or from a smaller corpus (e.g., a single composer or a single piece) that constrains the improvisation.

Several systems have been developed over the years for machine improvisation, focusing first on one-dimensional improvisation with one-dimensional training, using different methods from statistical sequence modeling such as compression-inspired incremental parsing (Dubnov, Assayag, and El-Yaniv 1998), Markovian models (Pachet 2002), and other machine-learning techniques (Conklin and Witten 1995; Dubnov et al. 2003). The use of a factor oracle from the field of string processing paved the way to the popular OMax interactive improvisation software (Assayag et al. 2006; Surges and Dubnov 2013). Several ideas have been spawned from the OMax project to approach the concept of polyphonic information in automatic improvisation. ImproteK (Nika, Chemillier, and Assayag 2017) has been developed for music based on temporal scenarios (e.g., a chord chart in jazz music). This system uses prior knowledge of a scenario that can represent dimensions other than the one being generated to guide the improvisation. Donze et al. (2013) use an automaton to control a melodic improvisation through rule-based grammars with information from other dimensions. In these examples, however, both the generated improvisations and the training are one-dimensional. Indeed, ImproteK focuses on co-occurrences between the generated dimension and the specific scenario, and Donze and coworkers assume manually specified rules, which do not generalize to other musical styles. The SoMax project (Bonasse-Gahot 2014) uses active listening over several dimensions to guide the improvisation. The views of each dimension activate

Computer Music Journal, 42:2, pp. 52–66, Summer 2018
doi:10.1162/COMJ_a_00460

places in the memory separately, however, and they do not consider the relations between the different dimensions. Training over several dimensions for one-dimensional generations has been studied for music analysis and automatic composition. Raczyński, Fukayama, and Vincent (2013) interpolate probabilistic models of melody, harmony, and tonality for a harmonization task. Methods using deep and/or recursive neural networks have also been used to create harmonizations (Bellgard and Tsang 1999) and to generate melodies over chord sequences (Bickerman et al. 2010). These systems are not constrained by a local frame, however, making it difficult to adapt to the particular style of a human musician in real time. More recently, multidimensional generation with multidimensional training has been studied. Padilla and Conklin (2016) generate counterpoint in the style of Palestrina with vertical viewpoints (cf. Conklin 2002) representing the correlation between two voices. Valle et al. (2016) propose an extension of the work by Donze and colleagues in which improvisation rules are obtained through data mining and could therefore be automatically adapted to different styles given a representative corpus. In these models they use multidimensional symbols, however, which raise overfitting issues and make it impossible to generate co-occurences of elements that were not seen in the training corpus. Methods using convolutional neural networks have also been used—for instance, to generate multidimensional music from raw audio (van den Oord et al. 2016) and by using symbolic data (Yang, Chou, and Yang 2017). Once again, however, these systems cannot adapt to a local frame.

In this article, we present two systems. First, we present a system using training over multidimensional sequences to guide its one-dimensional improvisation. Then, we introduce a system generating multidimensional improvisations based on multidimensional training. The first system was introduced by Déguernel, Vincent, and Assayag (2016), and it combines interpolated probabilistic models with a factor oracle. On the one hand, the interpolated probabilistic models enable the system to consider the correlations between dimensions and to benefit from advanced smoothing and optimization techniques. They represent the "cultural background" of the system and can be trained on different corpora. On the other hand, the factor oracle represents the local frame of the improvisation, as is usual in OMax. This enables the system to consider a context of variable length, similar to variable Markov models (Wang and Dubnov 2014), usually longer than the one represented by the probabilistic model. These systems also benefit from the expertise of the heuristics developed for navigating the factor oracle in OMax (Assayag and Bloch 2007). By combining these two aspects, we are able to create improvisations following the logic of a local frame enlightened by global multidimensional knowledge. In this article we describe an extension to our work on the first system by conducting an evaluation with listening sessions with professional improvisers. The second system uses several agents communicating through a cluster graph via message-passing (Koller and Friedman 2009). Probabilistic graphical models have proven to be an efficient representation for the communication between musicians during a situation of free improvisation (Kalonaris 2016), but they have not yet been used for multiagent music generation. Each agent represents either a dimension or a musician and is represented by both a cultural background and a local frame. The communication between agents makes them take a decision following their own logic and knowledge while being influenced by the other agents in an interactive way. This combination results in a multidimensional improvisation. This system is also evaluated by professional improvisers.

In the first section of this article, we review the theory behind probabilistic model interpolation and smoothing techniques, the factor oracle, and the heuristics used in OMax for navigation. In the second section, we introduce the system that combines probabilistic models with the factor oracle. We then introduce the use of a cluster graph for communication between agents (each represented, in our case, by one factor oracle). We first explain the theory of cluster graphs and the belief propagation algorithm, and then we propose a model combining a cluster graph and probabilistic

factor oracles. Finally, we present the results of our listening session for both systems.

Probabilistic Model Interpolation and Factor Oracle

We review here the different theoretical tools used by our systems based on the current state of the art. We first present the method of probabilistic model interpolation that enables our systems to take the multidimensional aspect of music into account. Then we present the factor oracle, the structure used in OMax that enables our systems to adapt to the local frame of an improvisation.

Probabilistic Model Interpolation

In the following, we adapt methods of probabilistic model interpolation for music generation. We first describe the global method, and then we present how the use of smoothing techniques can prevent overfitting when using training corpora of a limited size.

Method

Raczyński, Fukayama, and Vincent (2013) used probabilistic models for automatic harmonization on a corpus of classical music. We adapt this method for generating music. The goal is to create probabilistic models able to predict the evolution of one musical dimension using information from multiple dimensions. For instance, let us consider the problem of predicting the melody note M_t, encoded by pitch, played at time t. We want to estimate the probability $P(M_t|X_{1:t})$ where $X_{1:t}$ is a set of musical variables from various dimensions from time 1 to t. Such a model cannot be computed in practice, owing to its high combinatorics when using several dimensions on several time frames, the set of possibilities being the Cartesian product of the set of possibilities of each dimension in each time frame. Using probabilistic model interpolation enables us to consider several tractable submodels P_i, depending only on a subset of musical variables $A_{i,t} \subset X_{1:t}$ to approximate the global model. The interpolation can be linear (Jelinek and Mercer 1980):

$$P(M_t|X_{1:t}) = \sum_{i=1}^{I} \lambda_i P_i(M_t|A_{i,t}), \tag{1}$$

where I is the number of submodels and $\lambda_i \geq 0$ are the interpolation coefficients such that $\sum_{i=1}^{I} \lambda_i = 1$. The interpolation can also be log-linear (Klakow 1998):

$$P(M_t|X_{1:t}) = Z^{-1} \prod_{i=1}^{I} P_i(M_t|A_{i,t})^{\gamma_i}, \tag{2}$$

where $\gamma_i \geq 0$ are the interpolation coefficients and Z is the normalizing factor

$$Z = \sum_{M_t} \prod_{i=1}^{I} P_i(M_t|A_{i,t})^{\gamma_i}.$$

This method enables us to consider as many submodels as we want. For instance, a submodel can be a bigram $P(M_t|M_{t-1})$ predicting which melody note to play given the previous melody note, or a model representing "which melody note to play on which chord": $P(M_t|C_t)$. The chosen submodels are trained on a training corpus, and the probability of each submodel is estimated using a counting function over all the elements appearing in the corpus. Then the interpolation coefficients are estimated on a validation corpus to best approximate the global model. This estimation is done using the cross-entropy metric, which is equivalent in this case to the Kullback-Leibler divergence between the model and the validation corpus (Kullback and Leibler 1951), up to an additive constant:

$$H(M) = -\frac{1}{T} \sum_{t=1}^{T} \log_2 P(M_t|X_{1:t}), \tag{3}$$

where T is the number of time frames in the validation corpus. The cross-entropy represents the system's lack of understanding. Therefore, the interpolation coefficients are estimated to minimize the cross-entropy. The most relevant submodels will

be assigned large interpolation coefficients, whereas irrelevant submodels will receive interpolation coefficients close to zero.

Smoothing Techniques

When learning from a training corpus, it is common that all the observed elements in the training corpus do not include every single element that could appear during the test. This especially occurs when the training corpora are limited, which is usually the case for music improvisation, because corpora cannot be expected to reach the virtually infinite possibilities of a free improvisation. This leads to zero-value probabilities that can prevent some possible elements from being taken into consideration. Moreover, if the submodels chosen to represent the corpus are too complex, overfitting can occur. Smoothing techniques are used to correct the probabilities estimated from a limited corpus and to prevent overfitting. Plenty of smoothing techniques have been created to fit various applications. The following two techniques are among the most popular (Chen and Goodman 1998):

1. *Additive smoothing*: We consider that every possible element appears δ times more than it actually appears in the corpus.

$$P_{\text{add}}(X|Y) = \frac{\delta + \text{count}(X, Y)}{\sum_{X'} \delta + \text{count}(X', Y)}, \tag{4}$$

   where X is the event, Y is the context, and count is the function counting the number of times an element appears in the corpus (in this case, actually a pair of elements). This smoothing enables the model to overcome the problem of zero-value probabilities, because every element will appear at least δ times.
2. *Back-off smoothing*: We interpolate the model considered with a lower-order model.

$$P_{\text{back-off}}(X|Y) = \lambda P(X|Y) + (1 - \lambda)P(X|Z), \tag{5}$$

   where Z is a subset of Y and λ is the interpolation coefficient. For instance, if $P(X|Y)$ is an n-gram, then $P(X|Z)$ could be an $(n-1)$-gram. This smoothing enables the model to overcome the problem of overfitting. This smoothing technique can be used recursively. We can notice that back-off smoothing is actually a generalization of additive smoothing, because by recursion we always end up with a uniform distribution of all elements (0-gram).

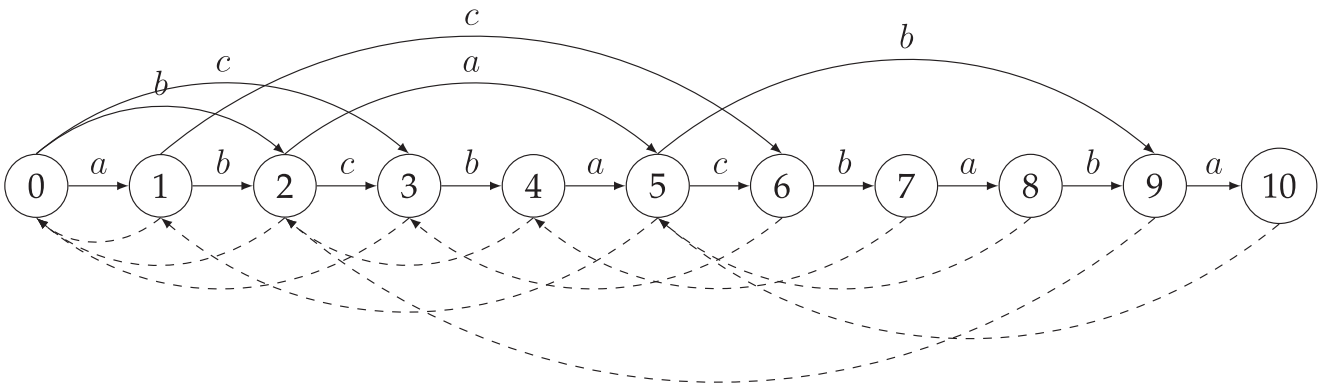
The use of probabilistic models enables us to take into consideration several dimensions and the correlation between them. When the models are used alone for generation, however, there is a lack of consistency, because there is no component enforcing some kind of repetition and local logic in the improvisation.

Factor Oracle in the OMax Paradigm

The factor oracle is a structure from the field of text algorithms, first introduced by Allauzen, Crochemore, and Raffinot (1999) for optimal string matching and then used by Lefebvre and Lecroq (2000) for computing repeated factors and data compression. It is an acyclic automaton representing at least all the factors in a word w. The construction algorithm is incremental and has complexity $O(|w|)$ in time and space. This structure was first adapted to music generation by Assayag and Dubnov (2004). An example of a factor oracle is shown in Figure 1 on the word $w = abcbacbaba$. This structure offers two main points of interest. First, it keeps the linear aspect of what is being learned. For instance, in Figure 1, we notice that the full word can be found following the horizontal arrows. Second, suffix links are created during its construction. These link locations in memory that have a similar context. For instance, in Figure 1, we can notice that the states 5 and 8, linked by a suffix link, share the context *cba*. The musical idea is that it is possible to jump from one point in memory to another one linked by a suffix link, thereby creating a new musical sentence but still preserving the musical style.

Assayag and Bloch (2007) developed heuristics for navigation in the factor oracle to create more realistic improvisations—for instance, by minimizing

Figure 1. Example of a factor oracle constructed on the word w = abcbacbaba. *Solid arrows are transitions and dashed arrows are suffix links. The suffix links connect each state to the leftmost previous state with which it shares the largest common context.*



the number of jumps through the use of a continuity factor, by avoiding loops through the use of a taboo list, etc. These heuristics also prevent the use of suffix links connecting states with a common context smaller than a fixed threshold. The factor oracle showed good results for modeling of improvisation style, and it has since been widely used in machine improvisation systems such as OMax, ImproteK, and PyOracle. This structure is not appropriate for multidimensional sequences, however. When considering several dimensions, the number of possible events is drastically increased (the alphabet would be the Cartesian product of the alphabet of each dimension). Therefore, locations in memory with a similar context would be rare, perhaps even nonexistent, limiting the generation to something extremely similar, or an exact replica of the memory, which would not be considered as an original improvisation.

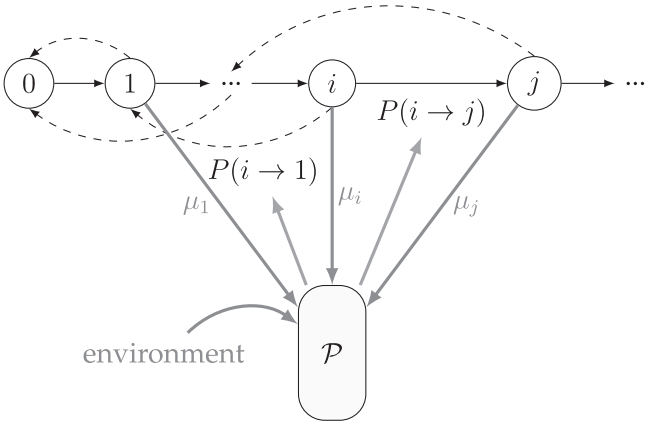
Factor Oracle Exploiting a Probabilistic Model

We introduce a system that creates improvisations in a way closer to that of a human improviser whose intuition of a context is enriched by knowledge and cultural background (Crispell 2000). The idea is to benefit both from the multidimensional training of probabilistic models and from the proficiency of the heuristics developed for the factor oracle, and for its extremely efficient scheme for incrementally building up a variable Markov type of linear memory.

On the one hand, a probabilistic model is created to represent the knowledge and cultural background of the musician we want to emulate. We select a set of submodels over the dimensions we want to take into consideration and apply interpolation and smoothing techniques to compose our global probabilistic model. This probabilistic model can be trained offline, prior to the performance, on a significant corpus representing the multidimensional knowledge acquired through our musical avatar's lifetime. On the other hand, during the performance we construct a factor oracle from a human musician's playing in a way similar to that of OMax, in an online fashion or from any reduced set of music, such as a single piece, following the dimension we want to generate (for instance, the melody). This constitutes the local frame of the improvisation.

We then generate the improvisation creating a path in the factor oracle as with OMax, except that we guide the improvisation using the probabilistic model. The factor oracle enforces the sequential logic and organic development of the motive being generated and enables the system to consider a longer context than the probabilistic model. This is made possible by the suffix links connecting each state with the previous state with the longest common context and by the heuristics developed in OMax, which ensures the use of suffix links connecting states with at least a minimal common context. The probabilistic model provides a deeper knowledge of music, owing to its training on a larger corpus, and the model enables the system

Figure 2. Using a multidimensional probabilistic model $\mathcal{P}$ with a factor oracle. Let us consider that from state i*, the only reachable states are state* j *and state 1. Using the context, μ_1, and μ_i, $\mathcal{P}$ is able to compute a score for the transition from state* i *to 1, similarly, for the transition from state* i *to* j *using the context, μ_i and μ_j. The scores are then normalized to calculate* P(i → 1) *and* P(i → j).



to consider multidimensional information and reinforce higher-level structures, such as harmony, over purely sequential logic.

At each step of the navigation, if we are in state i of the factor oracle, we compute the set of attainable states Att(i) considering the heuristics developed by Assayag and Bloch (2007). Then, considering the musical contents of state i $\mu_i = \{\mu_i^M, \mu_i^C, \ldots\}$ (that is to say the set of musical variables stored in state i during the factor oracle construction—for instance, μ_i^M represents the melody note of state i and μ_i^C represents the chord of state i), the musical contents of all attainable states, and possibly some information from the environment, we compute a score for each potential transition corresponding to the interpolation of the smoothed submodels from the probabilistic model. These scores are then normalized by the sum of the scores for each attainable state, thereby obtaining transition probabilities. For instance, if we are generating the melody note M_t, for all $j \in \text{Att}(i)$, the transition probability from state i at time $t-1$ to state j at time t is

$$P(i \to j | X_{1:t}) = \frac{P(M_t = \mu_j^M | X_{1:t})}{\sum_{k \in \text{Att}(i)} P(M_t = \mu_k^M | X_{1:t})}. \quad (6)$$

Finally, for generation, we chose a transition at random using these transition probabilities. Figure 2 illustrates the process for one step. The decision process for the navigation in the factor oracle is therefore enriched by the cultural background encoded in the probabilistic model.

This system could therefore provide better guidance on the leading dimension with multidimensional information. For instance, it could help reducing the brutal density or intensity changes that can be sometimes heard in systems such as OMax, if these dimensions were to be considered. Moreover, with active listening, the multidimensional aspect of this system can be used to guide the improvisation with environmental information as in SoMax. It is also possible to consider the creation of hybrid musicians, combining knowledge and local context from different styles. An extension of this system could be to replace the probabilistic model with either a recurrent or a deep neural net (Eck and Lapalme 2008) to learn longer-term dependencies or relations between dimensions that are more intricate.

Cluster Graphs and Message Passing between Oracles

We now introduce a model in which several factor oracles can communicate through message passing. Each oracle can represent either a musical dimension or a musician. The main idea is to get closer to multiagent systems that are more representative of real scenarios with free collective improvisation. The method we develop could therefore be used to create a polyphonic or multidimensional improvisation—for instance, a multi-instrument improvisation, a florid counterpoint, or a duet of melody and accompaniment. In the case of a multi-instrument scenario, this could represent the interactions between musicians, all trying to anticipate what the others are going to play in order to guide their own logic in improvisation, to produce realistic collective music making. This could also represent the cognitive process of individual musicians playing over several dimensions, trying to figure out the best way to conduct their improvisation using knowledge from all these dimensions (e.g., by improvising simultaneously over the melodic and harmonic dimensions). The different oracles communicate with probabilistic messages giving information

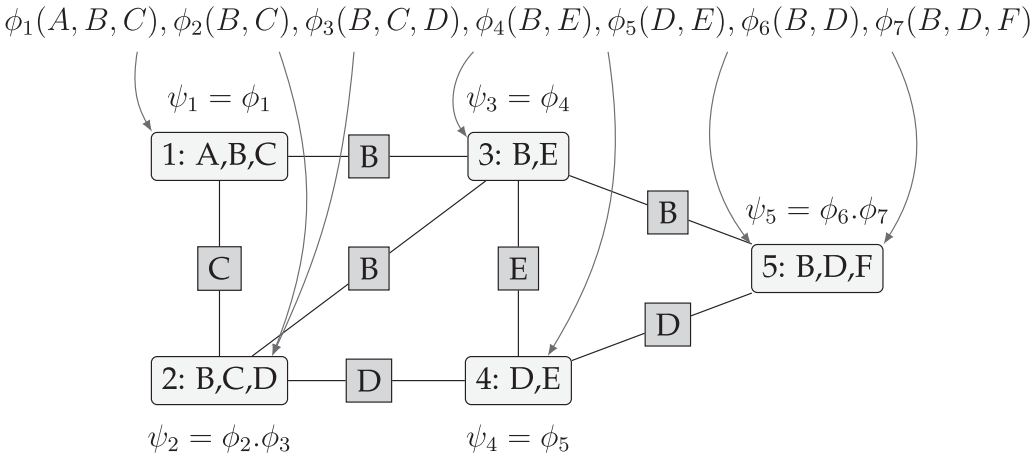


Figure 3. Example of factor distribution on a cluster graph. The clusters are represented by rounded rectangles, each consisting of a set of variables. The sepsets are represented by dark squares, each consisting of a subset of the common variables between the two clusters they connect. The clusters can communicate with their neighbors about the variables of their shared sepset.

about what they are about to do to inform the other oracles. This way every agent can make an informed decision accordingly. Message passing is organized on a graph representing which dimension each agent is working on and which dimensions it is listening to.

We chose to use belief propagation on a *cluster graph*, because it deals with inferring information with probabilistic models, and therefore is compatible with the system in the previous section. The main idea of this technique is that each agent has an initial belief based on its knowledge on a set of variables. These agents then communicate on some of the variables they share. Considering its initial belief and the information from the other agents, each agent can better estimate the marginal probability of its set of variables. We first present the theoretical tools needed to use the belief propagation algorithm, and then we present our method for multidimensional improvisation.

Cluster Graph and Message Passing

In this section we present the theoretical tools used to model the interaction between the factor oracles. We first present the cluster graph structure and its properties, and then we present the belief propagation algorithm. This section is a summary of the work on these tools presented by Koller and Friedman (2009).

Cluster Graph

Let X be a set of random variables. A *factor* ϕ is a function from $\mathrm{Val}(X)$ to $\mathbb{R}$. Note that this includes both joint probabilities and conditional probabilities. This will correspond to our submodels—for instance, a bigram on the melody $P(M_t|M_{t-1})$. The set of variables X is called the *scope* of the factor and written $\mathrm{Scope}[\phi]$. A cluster graph $\mathcal{U}$ for a set of factors Φ over a set of variables X is an undirected graph for which each vertex is associated with a subset of variables $\mathcal{K}_i \subseteq X$, called a *cluster*, and each edge between two clusters $\mathcal{K}_i$ and $\mathcal{K}_j$ is associated with a "separation set" or *sepset* $\mathcal{S}_{i,j} \subseteq \mathcal{K}_i \cap \mathcal{K}_j$—that is, a subset of variables shared by the two clusters about which they will communicate.

Considering a set of factors $\Phi = \{\phi_1, \ldots, \phi_k\}$, each ϕ_k is assigned to a cluster $\mathcal{K}_{\alpha(k)}$ such that $\mathrm{Scope}[\phi_k] \subseteq \mathcal{K}_{\alpha(k)}$. The initial belief of the cluster $\mathcal{K}_i$ is defined by

$$\psi(\mathcal{K}_i) = \prod_{k;\alpha(k)=i} \phi_k. \tag{7}$$

Figure 3 gives an example of how to distribute factors on a cluster graph. Note that other distributions could have been chosen in this example. For instance, instead of ϕ_1, ϕ_2 could have been assigned to $\mathcal{K}_1$ instead of $\mathcal{K}_2$. In this case, we would have

$$\psi_1(A, B, C) = \phi_1(A, B, C).\phi_2(B, C) \quad \text{and}$$
$$\psi_2(B, C, D) = \phi_3(B, C, D).$$

A cluster graph must follow two properties, and the example in Figure 3 satisfies both:

1. *Family preservation*: For each factor $\phi_k \in \Phi$, there must be a cluster $\mathcal{K}_i$ such that $\text{Scope}[\phi_k] \subseteq \mathcal{K}_i$. This ensures that every factor can be assigned to a cluster and, more generally, that all the information we want to take into account can be included in the cluster graph.
2. *Running intersection property*: For each pair of clusters $(\mathcal{K}_i, \mathcal{K}_j)$ and any variable $A \in \mathcal{K}_i \cap \mathcal{K}_j$, there is a unique path between $\mathcal{K}_i$ and $\mathcal{K}_j$ on which every cluster and sepset includes A. This is equivalent to the fact that, for any variable A, the set of clusters and sepsets including A forms a tree. This property has two consequences. First, the existence of this path enables the information about A to travel to every cluster that includes A. Second, the uniqueness of this path prevents the situation where the information about A goes in a circle, spawning false rumors. For example, in Figure 3, although cluster 1 and cluster 2 both contain the variable B, the sepset $\mathcal{S}_{1,2}$ contains only C, omitting B to avoid a circular path that would include cluster 3.

Belief Propagation Algorithm

The belief propagation algorithm is based on probabilistic message passing between clusters. The message passed from cluster i to cluster j over the variables from the sepset $\mathcal{S}_{i,j}$ is written $\delta_{i\to j}(\mathcal{S}_{i,j})$ and defined by

$$\delta_{i\to j}(\mathcal{S}_{i,j}) = \sum_{\mathcal{K}_i - \mathcal{S}_{i,j}} \psi_i \prod_{k\in(N_i - \{j\})} \delta_{k\to i}, \qquad (8)$$

where N_i is the neighborhood of i (i.e., the set of clusters that share a sepset with $\mathcal{K}_i$).

For instance, in Figure 3, the messages passed between clusters 1 and 3 are

$$\delta_{1\to 3}(B) = \sum_{A,C} \psi_1(A, B, C)\delta_{2\to 1}(C) \quad \text{and}$$

$$\delta_{3\to 1}(B) = \sum_{E} \psi_3(B, E)\delta_{2\to 3}(B)\delta_{4\to 3}(E)\delta_{5\to 3}(B).$$

Note that $\delta_{i\to j}(\mathcal{S}_{i,j})$ does not depend on $\delta_{j\to i}(\mathcal{S}_{i,j})$. This prevents a repetitive sending of the information received from a cluster back to the same cluster, which would result in the spawning of false rumors.

The belief propagation algorithm follows these steps:

1. Assign each factor ϕ_k in Φ to a cluster $\mathcal{K}_{\alpha(k)}$.
2. Compute the initial beliefs $\psi_i(\mathcal{K}_i) = \prod_{k:\alpha(k)=i} \phi_k$.
3. Initialize all messages to 1.
4. Repeat message updates following Equation 8.
5. Compute final beliefs $\beta_i(\mathcal{K}_i) = \psi_i \prod_{k\in N_i} \delta_{k\to i}$.

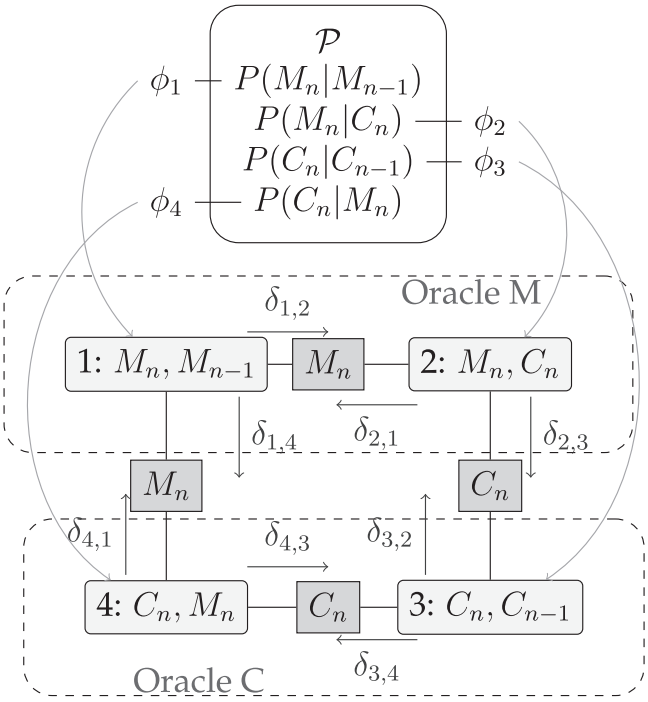
For a cluster, the final belief is a new factor based on its initial belief, updated by inference of the information from the other clusters. The term $\beta_i(\mathcal{K}_i)$ is an approximation of the marginal probability $P(\mathcal{K}_i)$.

The convergence of the belief propagation algorithm is not guaranteed for any cluster graph. Theoretically, the more complex the graph, the more complex convergence is. Note also that the order in which the messages are updated can have an influence on the convergence and on how fast it is. There is no way to determine the optimal order for message updates, however, this being completely dependent on the cluster graph construction. Synchronous message updates, where all messages are updated at the same time, have been proven to give the worst result in practice. In what follows, we have chosen to update messages in random order to avoid any bias. Even if theoretical convergence is not guaranteed, this algorithm shows good results in practice, except for very complex graphs with more than a thousand variables, which is not the case here (Koller and Friedman 2009).

Communication between Oracles for Improvisation

Our goal is to use the combination of smoothed submodels with the belief propagation algorithm on a cluster graph, to make several factor oracles

Figure 4. Cluster graph for multidimensional melody and harmony improvisation.



communicate with each other and thereby create a multidimensional improvisation in which several dimensions are generated at the same time. Each oracle represents a dimension or a musician and is trained on a context accordingly. The paths on the oracles are guided both by the probabilistic models defining the initial potentials and by the messages passed between oracles through the cluster graph. The idea is that the agents will try to find common ground, communicating to each other their musical expectations, considering their separate knowledge. This way, the oracles make a general choice of their paths from internal and external knowledge.

In Figure 4 we show the cluster graph we used to create an improvisation with both melodic and harmonic data. This could be used, for instance, to represent a pianist freely improvising both chords and melody, or to represent the interplay between a pianist and a saxophonist in the context of a free improvisation. We use n-gram models $P(M_n|M_{n-1})$ and $P(C_n|C_{n-1})$, for melody and for harmony, respectively, and models $P(M_n|C_n)$ and $P(C_n|M_n)$ to represent the direct relations between melody and harmony. Two oracles are constructed on the local context: Oracle M on melody and Oracle C on harmony. For each oracle, two clusters are created: one for the temporal aspect of the dimension, and another for direct relations between the two dimensions. Note that this cluster graph respects both the family preservation and the running intersection properties and is therefore suitable for the belief propagation algorithm.

At each step of the generation, each oracle provides its attainable states and its musical contents. The probabilistic model computes the factors ϕ_i corresponding to the smoothed submodels. This provides the initial potential for each cluster of the graph. The messages $\delta_{i,j}$ for the belief propagation are then all updated ten times, each time in a new random order. We then compute the final beliefs $\beta_i(\mathcal{K}_i)$ for each cluster. Therefore we can estimate $P(M_n)$ from $\beta_1(\mathcal{K}_1) = \beta(M_n, M_{n-1})$ or $\beta_2(\mathcal{K}_2) = \beta(M_n, C_n)$. For instance,

$$P(M_n) \simeq \sum_{C_n} \beta(M_n, C_n).$$

In theory, if the algorithm converged and produced exact inference, we would have

$$\sum_{C_n} \beta(M_n, C_n) = \sum_{M_{n-1}} \beta(M_n, M_{n-1}) = P(M_n).$$

On the other hand, we can similarly estimate $P(C_n)$ from $\beta_3(\mathcal{K}_3) = \beta(C_n, C_{n-1})$ or $\beta_4(\mathcal{K}_4) = \beta(C_n, M_n)$. The estimated $P(M_n)$ and $P(C_n)$ are normalized to obtain transition probabilities in Oracle M and Oracle C, respectively, as in the previous section. Each oracle then makes a decision regarding its own transitions following these transition probabilities.

This model can be extended to a higher number of dimensions or musicians, or to a higher number of submodels, as long as the constructed cluster graph follows the properties of family preservation and running intersection. Moreover, one of the main benefits of this method is that it would be possible to use several probabilistic models, one per oracle, trained on different corpora to emulate the styles of different musicians, creating an individuality for each agent and making this system more versatile

than a system that uses centralized knowledge with joint probabilities.

Experimentation

To evaluate the methods presented in this article we have generated improvisations using the *Charlie Parker Omnibook* (Parker and Aebersold 1978) as a corpus. This corpus consists of 50 tunes composed, played, and improvised on by Parker represented with symbolic melodic and harmonic data. This corpus can be found at http://repmus.ircam.fr/dyci2/ressources.

This bebop jazz musician has a fairly distinctive style and is therefore a good choice to assess the style modeling of our methods. We divided this corpus into three nonoverlapping subcorpora: a training corpus consisting of 40 tunes and improvisations, for training the different submodels; a validation corpus consisting of 5 tunes and improvisations, for optimizing the interpolation and smoothing coefficients; and a test corpus consisting of 5 tunes and improvisations, used to create the factor oracles during generation.

To qualitatively evaluate the generated improvisations, we conducted listening sessions with three professional jazz musicians: the saxophonist and jazz teacher Pascal Mabit, a graduate of the Conservatoire National Supérieur de Musique et de Danse de Paris; the double bassist Louis Bourhis, a graduate of the Haute École de Musique de Lausanne; and the pianist and jazz teacher Joël Gauvrit, a graduate of the Conservatoire National Supérieur de Musique et de Danse de Lyon. As professional jazz musicians they are familiar with the music of Charlie Parker and therefore able to provide us with valuable feedback.

Factor Oracle and Probabilistic Model

We now present the results of the two experiments performed with our first system. We first evaluate the impact of using a probabilistic model to guide a factor oracle. Then we evaluate the impact of the training corpus when guiding a factor oracle with a probabilistic model.

Guiding Improvisation with a Probabilistic Model

To evaluate our model with a factor oracle exploiting a probabilistic model, we conducted two experiments. First, we generated free improvisations, to compare improvisations generated by a factor oracle alone to ones generated by a factor oracle combined with a probabilistic model. For the latter, we chose two submodels:

1. a bigram on the melody $P_1(M_t|X_{1:t}) = P(M_t|M_{t-1})$ and
2. a model representing the correlations between melody and harmony $P_2(M_t|X_{1:t}) = P(M_t|C_t)$.

Those submodels and the interpolation and smoothing coefficients are trained on the *Omnibook* corpus, respectively, on the training corpus and the validation corpus. In this experiment, the harmony is not played, although the chord chart of the original tune is followed when generating an improvisation with the probabilistic model. We generated a dozen improvisations by both methods on two tunes: "Anthropology" and "Donna Lee." (The first of these uses "rhythm changes," i.e., the same chord progression as Gershwin's "I've Got Rhythm.") Examples of improvisations generated for this experiment are available at http://www.mitpressjournals.org/doi/suppl/10.1162/COMJ_a_00460.

After only a few examples, the three musicians noticed a clear difference between the two methods in the organization of the improvisation. With the first method, Bourhis said that the improvisations were "patchworks" of Parker elements without a feeling of consistency. Mabit added that with this method the harmonic progressions were not clear or were arranged in a random way, except when the improvisation consisted of direct quotes from the theme:

> Harmony makes sense in a continuity. . . . At the moment, it doesn't take that into account, or it is juxtaposing them in a random manner.

> We don't really hear harmony. We hear note after note, or phrases after phrases. And even inside phrases, there is not necessarily any harmonic sense.

When using a probabilistic model, both Bourhis and Mabit were able to say which chords the improvisation was playing on, despite them not actually being played. Moreover, Mabit found that there was a clear sense of the succession of tonal centers. Despite that, the improvisation preserves the global style of Charlie Parker thanks to the local context provided by the factor oracle. On "Donna Lee," Bourhis underlined that this harmonic clarity made the improvisation easier to follow and it would be easier to accompany:

> We feel much more at home when we hear that. . . . For instance, you can clearly hear the modulation to the fourth degree or the relative on the two places where they are characteristic. It does it the right way. We hear that it follows something. So it is much easier to understand.

Gauvrit, focusing more on the melodic phrases, noticed an improvement on the organization of the sequence of phrases when using the probabilistic model. The phrases feel less disjointed and more structured:

> I feel like the elements are more developed, that there is more unity. . . . It feels like there is an idea being developed—not globally, but something that sounds like reality, where there is an idea that brings another one, which brings another, and so on.

Gauvrit and Mabit noticed, however, that on top of some harmonic mistakes, there are still some hazy moments in the improvisation, especially on the bridge of "Anthropology," owing to a lack of understanding of the global form of the chord chart. More generally, the improvisations make sense from a harmonic point of view on a local level, but lack construction and logic with regard to the position in the chord chart. This comment was expected, because this problem exists in every system in the OMax paradigm and our method did not intend to solve this particular problem. Bourhis commented that,

> When it will understand the idea of global form, it will be even better, because at the moment, I feel as if it just takes the chords one after the other. . . . What it does works with the chords but it doesn't always make sense.

Moreover, Bourhis regretted the lack of harmonic anticipation and melodic leading to the future chord, saying that the improviser "knows what it is doing but not where it is going." This comment was also expected, since no anticipatory methods were implemented.

This first experiment showed good results overall. The impact of a probabilistic model can be noticed by professional jazz musicians, and the generated improvisations that use one are preferred over those generated from a factor oracle alone.

About the Corpus Choice

We then conducted a second experiment to see if differences could be heard when using probabilistic models trained on different corpora. We generated several improvisations on "Anthropology" and "Donna Lee" without any rhythmic information (only quarter notes and quarter rests were played) to avoid rhythmic offsetting (which would occur, for instance, when playing only two thirds of a triplet) on the respective chord charts that were now being played along with the improvisation to highlight the relations between melody and harmony. We first generated improvisations using the *Omnibook* corpus for training and then using a training corpus consisting of about one thousand pieces of classical music instead. The factor oracles are in both cases constructed on Charlie Parker's tune. Examples of generations for this experiment are available at http://www.mitpressjournals.org/doi/suppl/10.1162/COMJ_a_00460.

First, Mabit noticed that, with both methods, the global idea of Charlie Parker's style is still present, even when using the corpus drawn from classical music. This can be explained by the dominance of the local context provided by the factor oracle. After more listening, however, he pointed out

that, when using the classical music corpus, the improvisations seemed to aim more for the notes in the chords than when using the *Omnibook* corpus. The improvisations seemed more careful, and therefore sounded better from a harmonic point of view.

> The most credible method in my opinion is the one with the classical music corpus. It works better because there is a better consideration of the harmonic spaces, it takes more into consideration what is going on on each chord. . . . It sounds like someone who plays with the harmony and takes some liberties.

Bourhis explained the difference by saying that the improvisations generated with the classical music corpus are more strict from a harmonic point of view but less representative of Parker's style from a melodic viewpoint. Gauvrit underlined the difference between the two methods in a similar way, saying that when using the classical music corpus, the improvisations sounded "less altered" and sometimes even "Broadway-like":

> There is less harmonic inconsistency with the classical music corpus. With the *Omnibook*, there are quite a few things that sound out, a bit twisted, but at the same time, that's what makes them sound more like jazz. . . . It's less surprising with the classical corpus, it's more square, more academic.

The results of this experiment are encouraging, and the three musicians were able to notice a difference when using different corpora. The preferred corpus depends, however, on the esthetics and personal tastes of the individual musicians. Mabit tended to prefer the classical music corpus, whereas Bourhis and Gauvrit preferred the improvisations generated using the *Omnibook*.

Cluster Graph and Communication

To evaluate our interactivity model with a cluster graph and message passing, we used the cluster graph previously shown in Figure 4 to generate both melody and harmony. Once again, no rhythmic information was considered for the melody, which plays only quarter notes and quarter rests. The probabilistic model was trained on the *Omnibook* corpus. We generated multidimensional improvisations on "Anthropology" and "Donna Lee," on which the melodic and harmonic factor oracles were constructed. Both dimensions were played and improvised. Examples of generations for this experiment are available at http://www.mitpressjournals.org/doi/suppl/10.1162/COMJ_a_00460.

First of all, the three musicians praised the logic of the generated harmonic progressions, saying that it worked in all the examples generated, sounded like a real jazz song, and could have easily been played upon. Mabit thought that the generated improvisations were quite realistic and could even represent a real-life situation:

> It's funny, it really sounds like a wacky idea from the CNSM [Conservatoire National Supérieur de Musique] experimental improvisation class. Like, we work one month on "Donna Lee," just "Donna Lee," and now we know the chords and play "Donna Lee" but in an unstructured way.

The three musicians also said that the relations between the two dimensions made sense overall. Gauvrit and Bourhis raised the same criticism from the previous experiences that were made about the melody regarding the lack of global organization. Additionally, Mabit noticed that the melody followed the harmony properly, but might be too subordinated to it, and so he was less convinced by the generated melody, which felt a bit bland at times and was not sufficiently reactive:

> It seems like the two voices kind of know, or exactly know, what is going on with each other at all time, so it is the point where they know too much and it restricts them.

Generally, the generated multidimensional improvisations seemed quite realistic and musical. Even if sometimes feeling a bit too constrained and lacking global organization, this system improvises over several dimensions with both a horizontal and a vertical logic, providing encouraging results.

Discussion and Conclusion

We have presented two methods that are able to learn multidimensional information to generate musical improvisations.

First, we have shown the musical potential of combining probabilistic models with a factor oracle to guide the improvisation. The probabilistic models provide an efficient way to represent the relation between dimensions, and they can benefit from advanced smoothing techniques and optimization for interpolation, making them an efficient and comprehensive way to model the cultural background of a musician. The factor oracle is a structure that exploits efficient heuristics to represent the local context and the logic behind the development of a motive played by a musician. Therefore the proposed method is able to follow the contextual logic of an improvisation while enriching its musical discourse from multidimensional knowledge in a way that is closer to that of a human improviser.

Second, we introduced a method that models the interactivity between several musicians, or between several dimensions in an improviser's mind. This method is able to generate actual multidimensional improvisations. The communication between agents is conducted via a cluster graph. Smoothed probabilistic models are used as prior knowledge, and a belief propagation algorithm with message passing is used. Once again, the local context of each dimension is represented by a factor oracle. This way each agent is able to make a global decision regarding its own generation using both internal and external knowledge.

Both methods were evaluated during listening sessions with professional jazz musicians. Both methods received good feedback overall and seemed to be able to generate quite realistic improvisations. Some limitations of the current status of these methods were raised during the listening sessions, especially about the lack of a global form for the melodic improvisations. This could be studied, for instance, with the use of recurrent neural networks for the probabilistic aspect (Eck and Lapalme 2008) or with a generative grammar describing the multiscale organization of the improvisation for the deterministic aspect (Lerdahl and Jackendoff 1983; Chomsky 1996).

These methods could be adapted to work with other existing improvisation systems, such as ImproteK, PyOracle, etc., to improve their results. It would also be interesting to compare the multiagent system and message passing with formalisms other than the belief propagation—for instance, with gossip algorithms (Vanhaesebrouck, Bellet, and Tommasi 2017). This work also opens the door for musicological research towards creating more realistic avatars of musicians, by trying to determine the influences of human musicians we want to emulate and by training probabilistic models on a corpus comprising these influences, while focusing generation on these musicians' own music to create the factor oracle.

Acknowledgments

This research is made possible by support from the French National Research Agency, in the framework of the project DYCI2 "Creative Dynamics of Improvised Interaction," ANR-14-CE24-0002-01 (http://repmus.ircam.fr/dyci2/project), and with the support of the Région Lorraine.

References

Allauzen, C., M. Crochemore, and M. Raffinot. 1999. "Factor Oracle: A New Structure for Pattern Matching." In G. Tel et al., eds. *SOFSEM'99: Theory and Practice of Informatics*. Berlin: Springer, pp. 291–306.

Assayag, G., and G. Bloch. 2007. "Navigating the Oracle: A Heuristic Approach." In *Proceedings of the International Computer Music Conference*, pp. 405–412.

Assayag, G., and S. Dubnov. 2004. "Using Factor Oracles for Machine Improvisation." *Soft Computing* 8(9):604–610.

Assayag, G., et al. 2006. "OMax Brothers: A Dynamic Topology of Agents for Improvization Learning." In *Proceedings of the ACM Workshop on Audio and Music Computing for Multimedia*, pp. 125–132.

Bellgard, M. I., and C. P. Tsang. 1999. "Harmonizing Music the Boltzmann Way." In N. Griffith and P. M. Todd, eds.

Musical Networks. Cambridge, Massachusetts: MIT Press, pp. 261–277.
Bickerman, G., et al. 2010. "Learning to Create Jazz Melodies Using Deep Belief Nets." In *Proceedings of the International Conference on Computational Creativity*, pp. 228–236.
Bimbot, F., et al. 2014. "Semiotic Description of Music Structure: An Introduction to the Quaero/Metiss Structural Annotations." In *Proceedings of the AES International Conference on Semantic Audio*, pp. 32–43.
Bonasse-Gahot, L. 2014. "An Update on the SOMax Project." Internal report. Paris: Institut de Recherche et Coordination Acoustique/Musique, Sciences et Technologies de la Musique et du Son.
Chen, S. F., and J. Goodman. 1998. "An Empirical Study of Smoothing Techniques for Language Modeling." Technical Report TR-10-98. Cambridge, Massachusetts: Harvard University, Computer Science Group.
Chomsky, N. 1996. *Studies on Semantics in Generative Grammar*. Berlin: de Gruyter.
Conklin, D. 2002. "Representation and Discovery of Vertical Patterns in Music." In *Proceedings of the International Conference of Music and Artificial Intelligence*, pp. 32–42.
Conklin, D., and I. H. Witten. 1995. "Multiple Viewpoint Systems for Music Prediction." *Journal of New Music Research* 1(24):51–73.
Crispell, M. 2000. "Elements of improvisation." In J. Zorn, ed. *Arcana: Musicians on Music*. New York: Granary, pp. 190–192.
Déguernel, K., E. Vincent, and G. Assayag. 2016. "Using Multidimensional Sequences for Improvisation in the OMax Paradigm." In *Proceedings of the Sound and Music Computing Conference*, pp. 117–122.
Donze, A., et al. 2013. "Control Improvisation with Application to Music." Technical Report UCB/EECS-2013-183. Berkeley: University of California, Department of Electrical Engineering and Computer Sciences.
Dubnov, S., G. Assayag, and R. El-Yaniv. 1998. "Universal Classification Applied to Musical Sequences." In *Proceedings of the International Computer Music Conference*, pp. 332–340.
Dubnov, S., et al. 2003. "Using Machine-Learning Methods for Musical Style Modeling." *IEEE Computer* 10(38):73–80.
Eck, D., and J. Lapalme. 2008. "Learning Musical Structure Directly from Sequences of Music." Technical Report 6128. Montreal: University of Montreal, Department of Computer Science.
Jelinek, F., and R. L. Mercer. 1980. "Interpolated Estimation of Markov Source Parameters from Sparse Data." In E. S. Gelsema and L. N. Kanal, eds. *Pattern Recognition in Practice*. Amsterdam: North Holland, pp. 381–397.
Kalonaris, S. 2016. "Markov Networks for Free Improvisers." In *Proceedings of the International Computer Music Conference*, pp. 181–185.
Klakow, D. 1998. "Log-Linear Interpolation of Language Models." In *Proceedings of the International Conference on Spoken Language Processing*, pp. 1695–1698.
Koller, D., and N. Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, Massachusetts: MIT Press.
Kullback, S., and R. Leibler. 1951. "On Information and Sufficiency." *Annals of Mathematical Statistics* 22(1):79–86.
Lefebvre, A., and T. Lecroq. 2000. "Computing Repeated Factors with a Factor Oracle." In *Proceedings of the Australasian Workshop On Combinatorial Algorithms*, pp. 145–158.
Lerdahl, F., and R. Jackendoff. 1983. *A Generative Theory of Tonal Music*. Cambridge, Massachusetts: MIT Press.
Nika, J., M. Chemillier, and G. Assayag. 2017. "ImproteK: Introducing Scenarios into Human–Computer Music Improvisation." *ACM Computers in Entertainment* 14(2):Art. 4.
Pachet, F. 2002. "The Continuator: Musical Interaction with Style." In *Proceedings of the International Computer Music Conference*, pp. 211–218.
Padilla, V., and D. Conklin. 2016. "Statistical Generation of Two-Voice Florid Counterpoint." In *Proceedings of the Sound and Music Computing Conference*, pp. 380–387.
Parker, C., and J. Aebersold. 1978. *Charlie Parker Omnibook*. Los Angeles: Alfred Music.
Raczyński, S. A., S. Fukayama, and E. Vincent. 2013. "Melody Harmonization with Interpolated Probabilistic Models." *Journal of New Music Research* 42(3):223–235.
Surges, G., and S. Dubnov. 2013. "Feature Selection and Composition Using PyOracle." In *Proceedings of the International Workshop on Musical Metacreation*, pp. 114–121.
Valle, R., et al. 2016. "Specification Mining for Machine Improvisation with Formal Specifications." *Computers in Entertainment* 14(3):Art. 6.
van den Oord, A., et al. 2016. "Wavenet: A Generative Model for Raw Audio." Available online at arxiv.org/pdf/1609.03499.pdf. Accessed January 2018.

Vanhaesebrouck, P., A. Bellet, and M. Tommasi. 2017. "Decentralized Collaborative Learning of Personalized Models over Networks." In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 509–517.

Wang, C., and S. Dubnov. 2014. "Guided Music Synthesis with Variable Markov Oracle." In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pp. 55–62.

Yang, L., S. Chou, and Y. Yang. 2017. "MidiNet: A Convolutional Generative Adversarial Network for Symbolic-Domain Music Generation." In *Proceedings of the International Conference on Music Information Retrieval*, pp. 324–331.