Projet DYCI2,
WP2 Apprentissage interactif de structures musicales,
WP2.1 Apprentissage de structures multi-dimensionnelles

ANR-14-CE24-0002-01

Rapport de livrable :

L2.1.2 Apprentissage de structures multi-dimensionnelles, version finale algorithme.

Livrable	Date	Contributeurs	Rédacteurs	Contenu
L2.1.2 Version 01	Septembre 2017	K. Déguernel, E. Vincent (Inria), G. Assayag (Ircam-STMS)	E. Vincent	Maquette Logicielle, Rapport scientifique

Résumé

Ce document décrit l'évaluation expérimentale de deux des contributions effectuées dans la thèse de Ken Déguernel et décrites dans le livrable L2.1.1 : a) une nouvelle méthode d'improvisation qui combine une mémoire musicale probabiliste apprise sur un corpus de morceaux et un oracle des facteurs déterministe qui représente l'improvisation en cours ; c) une méthode de communication entre différents systèmes d'improvisation basée sur la théorie du « message passing » probabiliste.

Adresse du livrable logiciel

DYCI2_WP2_L2.1.2.zip
sur
https://forge.ircam.fr/p/Dyci2/

Parcours de l'oracle des facteurs orienté par un modèle probabiliste

Cette méthode est décrite dans les parties 2 et 3 du livrable L2.1.1. Afin de l'évaluer, nous avons généré différentes improvisations unidimensionnelles à partir d'oracles mélodiques construits chacun sur un morceau parmi 5 morceaux de Charlie Parker :

- des improvisations de type OMax sans aucune utilisation de module probabiliste,
- des improvisations avec un module probabiliste appris sur un corpus de 50 improvisations de Charlie Parker,
- des improvisations avec un module probabiliste appris sur un corpus de musique classique.

Nous avons recueilli et analysé l'avis de Pascal Mabit, musicien professionnel de jazz, sur ces improvisations. D'après lui, l'impact du module probabiliste est audible et les improvisations générées avec ce module sont préférées aux autres. Des tests d'écoute supplémentaires sont en cours avec deux autres musiciens professionnels. Ce travail a donné lieu aux publications [1,2].

Communication entre oracles par *belief propagation*

Cette méthode est décrite dans la partie 4 du livrable L2.1.1. Afin de l'évaluer, nous avons généré différentes improvisations multidimensionnelles à partir d'oracles mélodiques et harmoniques appris sur les 5 mêmes morceaux de Charlier Parker et communicant entre eux par propagation de croyances. Pascal Mabit a jugé ces improvisations réalistes et de bonne qualité musicale. Des tests d'écoute supplémentaires sont en cours avec deux autres musiciens professionnels. Ce travail est également décrit dans la publication [2].

Références

[1] Ken Déguernel, Emmanuel Vincent, Gérard Assayag, "Using multidimensional sequences for improvisation in the OMax paradigm", in *Proc. 13th Sound and Music Computing Conference*, Aug 2016.

[2] Ken Déguernel, Emmanuel Vincent, Gérard Assayag, "Probabilistic factor oracles for multidimensional machine improvisation", accepté moyennant modifications mineures dans *Computer Music Journal*.

Using Multidimensional Sequences For Improvisation In The OMax Paradigm

Ken Déguernel
Inria
STMS Lab, Ircam / CNRS / UPMC
ken.deguernel@inria.fr

Emmanuel Vincent
Inria
emmanuel.vincent@inria.fr

Gérard Assayag
STMS Lab, Ircam / CNRS / UPMC
gerard.assayag@ircam.fr

ABSTRACT

Automatic music improvisation systems based on the OMax paradigm use training over a one-dimensional sequence to generate original improvisations. Different systems use different heuristics to guide the improvisation but none of these benefits from training over a multidimensional sequence. We propose a system creating improvisation in a closer way to a human improviser where the intuition of a context is enriched with knowledge. This system combines a probabilistic model taking into account the multidimensional aspect of music trained on a corpus, with a factor oracle. The probabilistic model is constructed by interpolating sub-models and represents the knowledge of the system, while the factor oracle (structure used in OMax) represents the context. The results show the potential of such a system to perform better navigation in the factor oracle, guided by the knowledge on several dimensions.

1. INTRODUCTION

Current automatic music improvisation systems such as OMax [1] are able to learn the style of a one-dimensional musical sequence (a melody represented by a sequence of pitches or timbral audio features) in order to generate original improvisations by recombining the musical material. This style modeling can be performed live from a musician's playing or offline with a corpus. Several systems have been developed over the years using statistical sequence modeling [2], Markovian models [3] and other machine learning techniques [4]. However, most of these systems do not take the correlations between several musical dimensions (pitch, harmony, rhythm, dynamic, timbre...) into account.

Taking into consideration multiple dimensions and the relations between them has been an issue for systems out of the OMax paradigm. ImproTek [5, 6] makes use of a prior knowledge of a scenario (for example a chord chart) to guide the improvisation. SoMax [7] uses an active listening procedure enabling the system to react to its environment by activating places in its memory. PyOracle [8] uses information dynamics on audio features to create improvisations. Donze et al. [9] use an automaton in order to control the melodic improvisation with information about other dimensions. But in all of these, the actual training is still done on a one-dimensional sequence.

Training on multidimensional sequences has been studied by Conklin et al. [10] with multiple viewpoint systems where different attributes of a melody (such as pitches, intervals, contour...) are linked together for melody prediction on Bach chorales. These systems have also been studied for four part harmonisation [11]. Raczyński et al. use interpolated probabilistic models to do melody harmonisation [12]. This work proposes a flexible way to create a global model from chosen sub-models whose weight can be optimised and can be used in practice since the size of the model is reduced in order to learn the dependencies between dimensions. This method also uses smoothing techniques [13] to reduce overfitting issues that would otherwise arise. Some multidimensional models based on deep neural networks have also been proposed for the harmonisation problem [14] or to create jazz melodies [15]. In this case, the dependencies between dimensions are implicitly represented in the hidden layers.

In this article we present a way to use interpolated probabilistic models to create improvisations taking into account multiple musical dimensions and the correlations between them while keeping the benefits of the OMax paradigm and its factor oracle based representation [16], in particular its linear time oriented graph structure and optimised navigation scheme that make it a proficient tool for improvised performance and interaction. These are well-established methods that can profit from advanced smoothing and optimisation techniques. Moreover, they provide more explanatory models than neural network and therefore can provide us a deeper insight into the studied musical style or the improviser's mind.
We combine these models with the factor oracle [17] structure used in OMax, thus creating a new system with a musical training, able to use prior multidimensional knowledge to guide itself in an improvisation context described by the factor oracle.

In section 2, we explain how interpolation of probabilistic models can be used to take multiple dimensions into account for melody generation. Then, in section 3, we introduce a system combining probabilistic models with the factor oracle. And finally, in section 4 we present some results of experimentations done with this new system.

2. INTERPOLATION OF PROBABILISTIC MODELS

2.1 Method

Our system relies on the work of Raczyński et al. in [12] on automatic harmonisation. We want to create a probabilistic model able to predict the melody given information from different musical dimensions. Let us denote by M_t the melody played at time t, represented by the pitch. We want to predict :

$$P(M_t|X_{1:t}) \tag{1}$$

where $X_{1:t}$ is a set of musical variables from times 1 to t. This model is able to take into account multiple musical dimensions since the musical variables included in $X_{1:t}$ can be from several dimensions.

However, the combinatorics behind such a model are too high, the set of possibilities being the cartesian product of the set of possibilites of each dimension. Therefore such a prediction cannot be used in practice. To make it applicable, we approximate this global model by interpolating several sub-models P_i, which are easier to compute, depending only a subset of the musical variables $A_{i,t} \subset X_{1:t}$. For instance, we can use an n-gram model over a single dimension, or models representing the direct interaction between dimensions, for example, "which note should I play at time t knowing the harmony at this time?".

The interpolation can be linear [18] :

$$P(M_t|X_{1:t}) = \sum_{i=1}^{I} \lambda_i P_i(M_t|A_{i,t}) \tag{2}$$

where I is the number of sub-models 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 [19] :

$$P(M_t|X_{1:t}) = Z^{-1} \prod_{i=1}^{I} P_i(M_t|A_{i,t})^{\gamma_i} \tag{3}$$

where $\gamma_i \geq 0$ are the interpolation coefficients and Z is a normalising factor :

$$Z = \sum_{M_t} \prod_{i=1}^{I} P_i(M_t|A_{i,t})^{\gamma_i}. \tag{4}$$

The optimisation over the interpolation coefficients enable the system to accept as many sub-models as possible. The most relevant sub-models will have a high interpolation coefficient while irrelevant sub-models will receive an interpolation coefficient close to zero. This could be extended with some sub-model selection similar to Model M [20].

Two methods of smoothing techniques are used, the latter being a generalisation of the former. [13].

- First we are going to use an additive smoothing which consist of considering that every possible element appears δ times more than it actually appears in the corpus, with usually $0 < \delta \leq 1$.

$$P_{\text{add}}(X|Y) = \frac{\delta + c(X,Y)}{\sum_{X'} \delta + c(X',Y)} \tag{5}$$

  where c is the function counting the number of times an element appears in the corpus. This smoothing enable the model to overcome the problem of zero probabilities which often occurs with small training corpora.

- Then, we are going to use a back-off smoothing which consist of using information from a lower order model.

$$P_{\text{back-off}}(X|Y) = \lambda P(X|Y) + (1-\lambda) P(X|Z) \tag{6}$$

  where Z is a subset of Y. For instance, if $P(X|Y)$ is a n-gram, then $P(X|Z)$ could be a $(n-1)$-gram. This smoothing enable the model to overcome the problem of overfitting

2.2 Application to improvisation

In order to test sub-model interpolation for melody generation, we have used a corpus of 50 tunes from the Omnibook [21] composed, played and improvised on by Charlie Parker. We divided this corpus into three sub-corpora:

- a training corpus consisting of 40 tunes and improvisations in order to train the different sub-models,
- a validation corpus consisting of 5 tunes and improvisations in order to optimise the interpolation and smoothing coefficients using cross-entropy minimisation,
- a test corpus consisting of 5 tunes and improvisations.

We decided to use two sub-models :

$$P_1(M_t|X_{1:t}) = P(M_t|M_{t-1})$$

$$P_2(M_t|X_{1:t}) = P(M_t|C_t)$$

where M_t represents the melody at time t, and C_t represents the chord label at time t.

We applied a combination of additive smoothing and back-off smoothing techniques using $P(M_t)$ as a lower order model. Therefore, for the linear interpolation, we have :

$$\begin{aligned} P(M_t|X_{1:t}) = {} & \alpha P(M_t) + \beta U(M_t) \\ & + \lambda_1 P(M_t|M_{t-1}) + \lambda_2 P(M_t|C_t) \end{aligned} \tag{7}$$

where α and β are the smoothing coefficients corresponding respectively to the back-off smoothing and additive smoothing, U is the uniform distribution and λ_1 and λ_2 are the interpolation coefficients. The conditional probabilities are estimated using the counting function c.

	coefficients				cross-entropy
	λ_1	λ_2	α	β	$H(M)$
B+M	**0.582**	**0.129**	**0.289**	**0**	**4.543**
B	0.672	0	0.328	0	4.572
M	0	0.639	0.361	0	4.881
U	0	0	0.998	0.002	5.858

Table 1. Cross-entropy results (bits/note) with linear interpolation. The results are shown for the smooth interpolation of the bigram model and melody/chord model (B+M), then for the bigram model with smoothing (B), then for the melody/chord model with smoothing (M), and finally with the smoothing alone (U) as a point of comparison.

In order to evaluate this model, we used the cross-entropy on the test corpus :

$$H(M) = -\frac{1}{T}\sum_{t=1}^{T}\log_2 P(M_t|X_{1:t}) \; . \qquad (8)$$

This metric is in this case equivalent to the KL-divergence up to an additive constant and represents the lack of understanding of the system. Therefore, the lower the cross-entropy, the better the model prediction power.

In Table 1, we present some of the results obtained with linear interpolation. Note that all the results are shown with the same smoothing technique in order to allow a proper comparison. As shown, the model has a better prediction power when using sub-model interpolation. However, the improvement is quite small in term of cross-entropy. This can be explained by the fact that the cross-entropy represents the system's ability to reproduce the test data, while improvisation is not about reproduction but about creativity, and as we said improvisation possibilities are unlimited.

However, informal listening tests show some improvement when using the interpolated model compared to a classic n-gram model. But generated improvisation with just this probabilistic model lack of consistency and of a local organisation. Therefore, we have decided to go further using this type of probabilistic model by combining them with the oracle factor.

3. FACTOR ORACLE EXPLOITING A PROBABILISTIC MODEL

The factor oracle is a structure coming from the field of bioinformatics and language theory [17, 22] that has been widely used in automatic improvisation systems such as OMax [1, 16], ImproTek [5], SoMax [7] or PyOracle [8]. This structure is able to keep the linear aspect of what is being learnt and create links, called suffix links, between places in the memory with a similar context. An example of factor oracle is shown Figure 1.

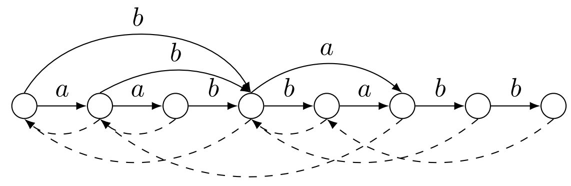

Figure 1. Example of factor oracle constructed on the word $w = aabbabb$. Horizontal solid arrows are the transition, bent solid arrows are the factor links and dashed arrows are the suffix links.

We designed a system combining the probabilitic model able to take into account the multidimensional aspect of music, with the contextual setting brought by the factor oracle. The idea was to conceive a system creating improvisation in a way closer to a human improviser. We were inspired by this quote from Marilyn Crispell's *Elements of Improvisation* [23] (written for Cecil Taylor and Anthony Braxton) :

> *The development of a motive should be done in a logical, organic way, not haphazardly (improvisation as spontaneous composition) – not, however, in a preconceived way – rather in a way based on intuition enriched with knowledge (from all the study, playing, listening, exposure to various musical styles, etc., that have occurred through a lifetime – including all life experiences); the result is a personal musical vocabulary.*

First, we create a probabilistic module with all the sub-models we want to take into consideration and the corresponding interpolation and smoothing coefficients necessary to the creation of the global probabilistic model. This module can be trained on a substantial corpus offline, but can also be trained (or updated) online with a musician's playing. In Crispell's quote, this matches with the knowledge acquired through the system's lifetime.

Second, we create an oracle factor for which the construction of states, edges and suffix links only depends on one dimension (usually the melody). The states can represent a single note as in OMax or a musical fragment (for instance a beat) as in ImproTek. In Crispell's quote, this correponds to the logic of the context in which the motive must be developed. The oracle is created online with a musician's playing, or with a corpus (usually smaller than the one used to create the probabilistic module).

The system is now able to improvise music, creating a path in the factor oracle that is guided and enriched by the knowledge from the probabilistic module. At each step, knowing the state the system is in, all the reachable states, and the musical contents in those states, we compute a score for each possible transition corresponding to the interpolation of the sub-models in the probabilistic module. Thus, we are enriching with external knowledge the decision of which edge to follow. We can then normalise the scores to obtain the probabilites of transitions and make a random choice following the resulting probabilities.

Let Att(i) be the set of reachable states from state i follow-

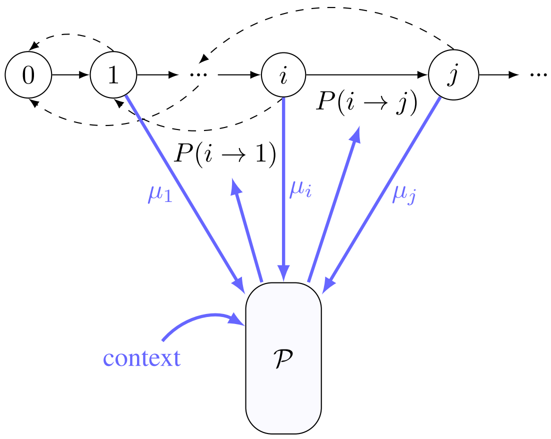

Figure 2. Using a multidimensional probabilistic model $\mathcal{P}$ with an oracle factor. Let us consider that from state i, the only reachable states are state j and state 1. Using the context, μ_1, and μ_i, $\mathcal{P}$ is able to compute a score for the transition from state i to 1. Same thing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

ing the heuristics explained in [16] (using suffix links and reverse suffix links for instance). Let $\mu_i = \{\mu_i^M, \mu_i^C, ...\}$ be the musical contents of state i, that is to say the set of musical variables stored in state i during the oracle construction (for instance, μ_i^M represents the musical content's melody of state i). Then, for all $j \in \text{Att}(i)$, the transition probability in the oracle from state i to state j, knowing the past context is :

$$P(i \to j|X_{1:t}) = \frac{P(M_t = \mu_j^M|X_{1:t})}{\sum\limits_{k \in \text{Att}(i)} P(M_t = \mu_k^M|X_{1:t})} \qquad (9)$$

In practice, for $X_{1:t}$, we use the musical contents from the previous and current states of the path of the factor oracle. Figure 2 illustrates this process for one step.

4. EXPERIMENTATION

To test the system proposed in the previous part, we generated some improvisations on Charlie Parker's music following three methods.

1. Some improvisations were made with OMax without any probabilistic module. The factor oracle was constructed on one tune (theme and Parker's improvisation).

2. Some improvisations were made with OMax with a probabilistic module. The sub-models considered are an n-gram model over the melody, and a relational model between melody and harmony. The probabilistic module was trained on Charlie Parker's whole Omnibook (50 themes and improvisations), and the factor oracle was constructed on one tune. The Omnibook corpus was created manually using MusicXML and includes both melodic information and chord labels. The idea here is to have a probabilistic module trained on a larger but similar corpus to the tune used for the factor oracle.

3. Some improvisations were made with OMax with a probabilistic module, similarly to the previous one, but the corpus used to train the probabilistic module is a classical music corpus of over 850 non improvised tunes while the factor oracle is constructed on a Charlie Parker tune (theme and improvisation). The classical music corpus was user-generated using MusicXML with both melodic and chord information and was screened for improper chord labels [12]. The idea here is to see how the system performs when trained on a corpus of a different style than the tune used for the factor oracle.

In the second and third method, the probalistic modules were trained using both melodic and harmonic information over all the tunes of each corpus. Three sub-models were used:

$$P_1(M_n|X_{1:n}) = P(M_n|M_{n-1})$$
$$P_2(M_n|X_{1:n}) = P(M_n|C_n)$$
$$P_3(C_n|X_{1:n}) = P(C_n|C_{n-1})$$

where n is an index over the note of the melody. M_n is the n^{th} notes of the melody, and C_n is the chord played over M_n.
Due to the nature of our dataset, we chose to use a small amount of sub-models and very simple one as a proof of concept. Better results would be expected with more sub-models (as mentioned in 2.1) but would require more complete data.

For each method, 15 improvisations were generated using 3 Charlie Parker tunes as reference : Au Private, Donna Lee and Yardbird Suite.
The generated improvisations can be listened online at `members.loria.fr/evincent/files/smc16`
and the MusicXML Omnibook corpus can be found at `members.loria.fr/evincent/files/omnibook`.

First of all, the most significant difference seems to be the harmonic stability appearing while using a probabilistic module trained with either the Omnibook or a classical music corpus. The improvisations generated using these methods seem to follow a harmonic framework, while the factor oracle is only constructed with the melody. For instance, this can be heard on the first example of Au Private. Second, when the probabilistic module is trained on a classical music corpus, while the harmonic stability is stronger, Charlier Parker's musical language looses its distinctiveness, as if the harmonic aspect was too strong a constraint. For instance, this can be noticed on the third example of Yardbird Suite. This comforts our initial idea that using a multidimensional training over an appropriate corpus enables our system to generate improvisations closer to a specific style.
Furthermore, according to listeners, the improvisations with a probabilistic module are more diverse, fluid and creative

than the simple oracle one. This is in part because the combination of dimensions and the smoothing provide escape mechanisms from usual mono-dimensional attractors (the obsessive jingle phenomenon due to high conditional probabilities and overfitting). For instance, this can be clearly heard in the first example of Donna Lee.

These results are encouraging. We only tested this system using melodic and harmonic relations, yet we can already hear a significant improvement on how the improvisations are guided through the factor oracle. This system could be extended to represent other interdimensional relations, in particular rhythm, beat phase and dynamic, with more detailed data from live playings, and therefore can be used for any style of music.
Moreover, this system's modularity makes it very adaptable, and could be integrated in other existing systems :

- A probabilistic module could be integrated in ImproTek [5], where the evolution of one dimension is predefined in a scenario. This would add some smoothing in ImproTek's improvisation and therefore expand its expressiveness.

- Similarly, a probabilistic module could be integrated in SoMax [7] where some of the context would come from active listening.

- Finally, this system could be adapted for PyOracle [8] using an interpolation where the dimensions are actually audio features.

5. CONCLUSIONS

We have shown the musical potentialities of the combination of probabilistic models with the factor oracle. This creates a system able to follow the contextual logic of an improvisation while enriching its musical discourse from multidimensional knowledge in a closer way to a human improviser. On the one hand, the probabilistic models enable the system to be trained on a multidimensional sequence and to take the relations between dimensions into account. They also profit from advanced smoothing and optimisation techniques which make them an efficient way to represent the musical knowledge acquired through a lifetime by a musician. On the other hand, the factor oracle is an efficient data structure able to represent the logic of a musical context. This system shows good potential to perform a better navigation in the factor oracle, generating improvisations closer to the desired style. Moreover, this system could be easily adaptated to other existing systems (ImproTek, SoMax, PyOracle...), potentially improving their results.

Acknowledgments

This work is made with the support of the French National Research Agency, in the framework of the project DYCI2 "Creative Dynamics of Improvised Interaction" (ANR-14-CE24-0002-01), and with the support of Region Lorraine.

6. REFERENCES

[1] G. Assayag and S. Dubnov, "Using factor oracles for machine improvisation," *Soft Computing*, vol. 8-9, pp. 604–610, 2004.

[2] D. Conklin, "Music generation from statistical models," in *Proceedings of the AISB Symp. on Artificial Intelligence and Creativity in the Arts and Sciences*, 2003, pp. 30–35.

[3] F. Pachet and P. Roy, "Markov constraints: steerable generation of Markov sequences," *Constraints*, vol. 16, no. 2, pp. 148–172, March 2011.

[4] S. Dubnov, G. Assayag, O. Lartillot, and G. Bejerano, "Using machine-learning methods for musical style modeling," *IEEE Computer*, vol. 10, no. 38, pp. 73–80, 2003.

[5] J. Nika and M. Chemillier, "Improtek, integrating harmonic controls into improvisation in the filiation of OMax," in *Proceedings of the International Computer Music Conference*, 2012, pp. 180–187.

[6] J. Nika, J. Echeveste, M. Chemillier, and J.-L. Giavitto, "Planning human-computer improvisation," in *Proceedings of the International Computer Music Conference*, 2014, pp. 330–338.

[7] L. Bonasse-Gahot, "An update on the SoMax project," IRCAM, Tech. Rep., 2014.

[8] G. Surges and S. Dubnov, "Feature selection and composition using pyoracle," in *Proceedings of the 2nd International Workshop on Musical Metacreation*, 2013.

[9] A. Donze, S. Libkind, S. A. Seshia, and D. Wessel, "Control improvisation with application to music," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2013-183, November 2013.

[10] D. Conklin and I. H. Witten, "Multiple viewpoint systems for music prediction," *Journal of New Music Research*, vol. 1, no. 24, pp. 51–73, 1995.

[11] R. P. Whorley, G. A. Wiggins, C. Rhodes, and M. T. Pearce, "Multiple viewpoint systems: Time complexity and the construction of domains for complex musical viewpoints in the harmonisation problem," *Journal of New Music Research*, no. 42, pp. 237–266, 2013.

[12] S. A. Raczyński, S. Fukayama, and E. Vincent, "Melody harmonisation with interpolated probabilistic models," *Journal of New Music Research*, vol. 42, no. 3, pp. 223–235, 2013.

[13] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," Harvard University, Tech. Rep. TR-10-98, 1998.

[14] M. I. Bellgard and C. P. Tsang, "Musical networks," N. Griffith and P. M. Todd, Eds. MIT Press, 1999, ch. Harmonizing Music the Boltzmann Way, pp. 261–277.

[15] G. Bickerman, S. Bosley, P. Swire, and R. M. Keller, “Learning to create jazz melodies using deep belief nets,” in *Proceedings of the International Conference on Computational Creativity*, 2010, pp. 228–236.

[16] G. Assayag and G. Bloch, “Navigating the oracle : A heuristic approach,” in *Proceedings of the International Computer Music Conference*, 2007, pp. 405–412.

[17] C. Allauzen, M. Crochemore, and M. Raffinot, “Factor oracle : A new structure for pattern matching,” *SOFSEM'99, Theory and Practice of Informatics*, pp. 291–306, 1999.

[18] F. Jelinek and R. L. Mercer, “Interpolated estimation of Markov source parameters from sparse data,” in *Pattern Recognition in Practice*, 1980, pp. 381–397.

[19] D. Klakow, “Log-linear interpolation of language models,” in *Proceedings of the 5th International Conference on Spoken Language Processing*, 1998, pp. 1695–1698.

[20] S. F. Chen, L. Mangu, B. Ramabhadran, R. Sarikaya, and A. Sethy, “Scaling shrinkage-based language models,” in *Proceedings of the Automatic Speech Recognition & Understanding*, 2009, pp. 299–304.

[21] C. Parker and J. Aebersold, *Charlie Parker Omnibook*. Alfred Music Publishing, 1978.

[22] A. Lefebvre and T. Lecroq, “Computing repeated factors with a factor oracle,” in *Proceedings of the 11th Australasian Workshop On Combinatorial Algorithms*, 2000, pp. 145–158.

[23] M. Crispell, “Elements of improvisation,” in *Arcana : Musicians on Music*, J. Zorn, Ed., 2000, pp. 190–192.

Probabilistic Factor Oracles for Multidimensional Machine Improvisation

Abstract

This paper presents two methods using training over multidimensional sequences for automatic improvisation. We first present a system combining interpolated probabilistic models with a factor oracle. The probabilistic models are trained on a corpus and provide information on the correlation between dimensions and are used to guide the navigation in the factor oracle that ensure a logical improvisation. The improvisation are therefore created in a way where the intuition of a context is enriched with multidimensional knowledge. We then introduce a system creating multidimensional improvisations based on interactivity between dimensions via message passing through a cluster graph. The communication infers some anticipatory behaviour on each dimension now influenced by the others, creating a consistent multidimensional improvisation. Both systems are evaluated by a professional improviser during listening sessions. Overall, they receive good feedback and show encouraging results, first on how a multidimensional knowledge can help performing better navigation in the factor oracle and second on how communication through message passing can emulate the interactivity between dimensions or musicians.

Introduction

Our goal is to design a system able to generate multidimensional musical improvisations. By "dimensions", we mean musical "layers" such as melody, harmony, rhythm, timbre, etc. [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 context 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 modelling such as compression-inspired incremental parsing [Dubnov et al. (1998)], Markovian models [Pachet (2002)] and other machine learning techniques [Dubnov et al. (2003)] or the use of a Factor Oracle structure from the field of string processing, paving the way to the popular OMax interactive improvisation software [Assayag and Dubnov (2004); Surges and Dubnov (2013)]. Several ideas have spawned around the OMax project [Assayag et al. (2006)] to approach the concept of polyphonic information in automatic improvisation. ImproteK [Nika and Chemillier (2012)] has been developed for music based on temporal scenarios (for instance a chord chart in jazz music). This system uses prior knowledge of a scenario that can represent another dimension 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. However, in these examples, the generated improvisations are still one-dimensional, and the training is also one-dimensional. Indeed, Improtek focuses on co-occurrences between the generated dimension and the specific scenario and [Donze et al. (2013)] assumes manually specified rules, which do not generalise to other musical styles.

Training over several dimensions for one-dimensional generations has been studied for music analysis and automatic composition. [Raczyński et al. (2013)] interpolate probabilistic models of melody, harmony and tonality for a harmonisation task. Methods using deep and/or recursive neural networks have also been employed to create harmonisation [Bellgard and Tsang (1999)] and melodies over chord sequences

[Bickerman et al. (2010)]. However, these systems are not constrained by a local frame.

More recently, multidimensional generation with multidimensional training has been studied for automatic composition. [Padilla and Conklin (2016)] generate counterpoint in the style of Palestrina with vertical viewpoints representing the correlation between two voices. [Van Den Oord et al. (2016)] use deep neural network to generate multidimensional music from raw audio. However, once again, these systems cannot adapt to a local context.

In this article, we propose two systems. First, we present a system using training over multidimentional sequences to guide its one-dimensional improvisation. Then, we introduce a system generating multidimensional improvisation with a multidimensional training. The first system was introduced in [Reference removed for anonymity(2016)] and 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 optimisation techniques. They represent the "cultural background" of the system and can be trained on different corpora. On the other hand, a factor oracle represents the local frame of the improvisation as per 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 n-grams and to benefit from the expertise of the heuristics developed for the navigation in 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 a global multidimensional knowledge. We extend this work by conducting an evaluation of this system with a listening session with a professional improviser. The second system uses several agents communicating through a cluster graph via message passing [Koller and Friedman (2009)]. Probabilistic Graphical Models have been proven to be an efficient representation for the communication between musicians during a situation of free improvisation by [Kalonaris (2016)] but have not yet been used for multi-agent music generation. Each agent represents either a dimension of 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, but influenced by the others in an interactive way. The combination of all agents results in a multidimensional improvisation. This system is also evaluated by a professional improviser.

In the first section we recall the theory behind probabilistic model interpolation and smoothing techniques, then talk about the factor oracle and the heuristics used in Omax for navigation. In the second section, we introduce the system combining probabilistic models with the factor oracle. In the third section, we introduce the use of a cluster graph for the communication between agents (represented in our case by factor oracles). We first explain the theory of cluster graphs and the belief propagation algorithm and then 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

Probabilistic model interpolation

Method

[Raczyński et al. (2013)] used probabilistic models for automatic harmonisation on a classical music corpus. We adapted this method for music generation ; the goal is to create probabilistic models able to predict the evolution of one musical dimension using information from various dimensions. For instance, let us consider the problem of predicting the melody M_t played at time t (encoded by the pitch). We want to estimate $P(M_t|X_{1:t})$ where $X_{1:t}$ is a set of musical variables from various dimensions from time 1 to t. Such a model can not be computed in practice due 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 sub-models P_i depending only on a subset of musical variables $A_{i,t} \subset X_{1:t}$ in order to approximate the global model. The interpolation can be linear [Jelinek and Mercer (1980)] in which case

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

where I is the number of sub-models 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)] in which case

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

where $\gamma_i \geq 0$ are the interpolation coefficients and Z is the normalising 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 sub-models as we want. The chosen sub-models are trained on a training corpus : the probabilities are estimated using a counting function over all the elements appearing in the corpus. Then the interpolation coefficients are optimised on a validation corpus in order to approximate at best the global model. The optimisation is done using the cross-entropy metric, equivalent in this case to the KL-divergence between the model and the validation corpus up to an additive constant :

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

Cross-entropy represents the lack of understanding of the system. Therefore, interpolation coefficients are optimised to minimise the cross-entropy. The most relevant sub-models will be assigned large interpolation coefficients while irrelevant sub-models will receive interpolation coefficients close to zero. Sub-model selection [Chen et al. (2009)] could be used in order to discard the irrelevant sub-models.

Smoothing techniques

When learning on a 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 where corpora cannot be expected to reach the virtually infinite possibilities of a free improvisation. This leads to some zero-value probabilities that can prevent some possible elements to be taken into consideration. Moreover, if the sub-models 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 prevent overfitting. Plenty of smoothing techniques have been created to fit best to various applications. The following two techniques are among the most popular [Chen and Goodman (1998)] :

- *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 count is the function counting the number of times an element (here, a pair of elements) appears in the corpus. This smoothing enables the model to overcome the problem of zero-value probabilities, everything appearing at least δ times.

- *Back-off smoothing :* we interpolate the considered model 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. 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 generalisation of additive smoothing, since by recursion we always end up with a uniform distribution of all elements (0-gram).

Using probabilistic models enable us to take into consideration several dimensions and the correlation between them. However when used alone for generation, there is a lack of consistence due to the fact that there is no component enforcing some kind of repetitition and local logic to the improvisation.

Factor Oracle in the OMax paradigm

The factor oracle is a structure from the field of bioinformatics and language theory first introduced by [Allauzen et al. (1999)] for optimal string matching and then used for computing repeated factors and data compression by [Lefebvre and Lecroq (2000)]. It is an acyclic automaton representing al least all the factors in a word w and for which the construction algorithm is incremental and $O(|w|)$ in time and space. This structure was first adapted to music generation by [Assayag and Dubnov (2004)]. An example of 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 learnt. For instance, in Figure 1, we can notice that the word can be found following the horizontal arrows. Second, suffix links are created during the construction of the automaton. These link places in the memory with 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 the memory to another one linked by a suffix link creating a new musical sentence but still preserving the musical style.

In [Assayag and Bloch (2007)], heuristics are developed for navigation in the factor oracle in order to create more realistic improvisation, with for instance the use of a continuity factor in order to avoid too many jumps, the use of a taboo list to avoid loops,

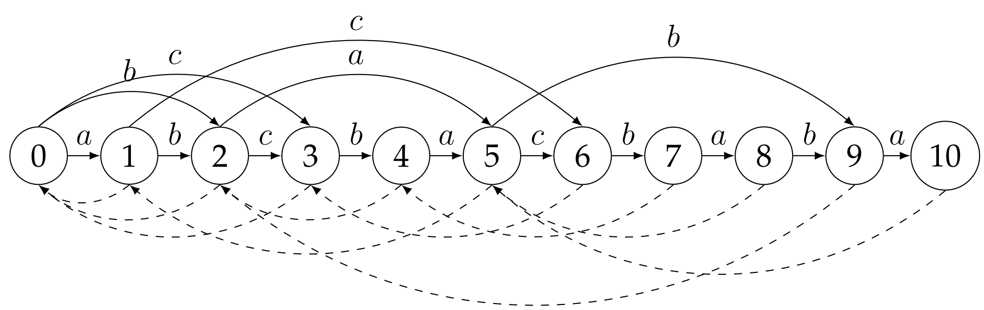

Figure 1. Example of factor oracle constructed on the word $w = abcbacbaba$. Solid arrows are the transitions and dashed arrows are the suffix links.

etc. The factor oracle showed good results for improvisation style modelling and has since been widely used in machine improvisation systems such as OMax, ImproteK or PyOracle. However, this structure is not appropriate for multidimensional sequences. When considering several dimensions, the amount of possible event is drastically increased (the alphabet would be the Cartesian product of the alphabet of each dimension). Therefore, places in the memory with a similar context would be rare, even perhaps inexistent, 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 creating improvisations in a closer way to a human improviser where the intuition of a context is enriched with knowledge and a cultural background [Crispell (2000)]. The idea is to benefit from both the multidimensional training of probabilistic models and the proficiency of the heuristics developed for the factor oracle and its extremely efficient scheme for incrementally build up a variable Markov type of linear memory.

On the one hand, a probabilistic module is created to represent the knowledge and cultural background of the musician we want to emulate. We select a set of sub-models over the dimensions we want to take into consideration and apply interpolation and smoothing techniques in order to compose our global probabilistic model. This probabilistic module can be trained offline, prior to the performance, on a significant corpus representing the multidimensional knowledge acquired through our musician avatar's lifetime.

On the other hand, during the performance, we construct in an online fashion a factor oracle in a similar way as OMax from a musician's playing 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 representation of the local context of the improvisation.

We then generate a machine improvisation creating a path in the factor oracle as with OMax except that we guide the improvisation using the probabilistic module. 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 module. This is thanks to the suffix links connecting each state with the previous state with the longest common context and to the heuristics developed in OMax ensuring the use of suffix links connecting states with at least a minimal common context. The probabilistic module provides a deeper knowledge of music, thanks to its training on a larger corpus, and enables the system to consider multidimensional information and re-enforce higher level structures such as harmony over the 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 $\text{Att}(i)$ considering the heuristics from [Assayag and Bloch (2007)]. Then considering the musical contents of state i $\mu_i = \{\mu_i^M, \mu_i^C, ...\}$, 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 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 sub-models from the probabilistic module. The scores are then normalised to obtain transition probabilities. For instance, if we are generating the melody 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 :

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

Finally, for generation, we chose the transition at random using those proper 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 module.

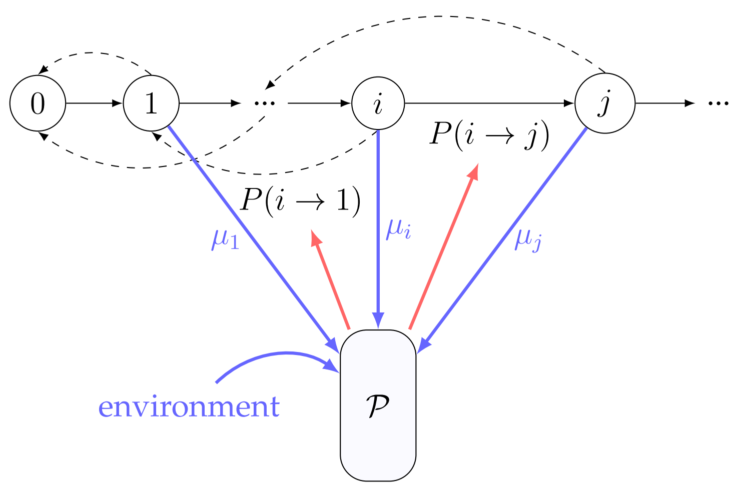

Figure 2. Using a multidimensional probabilistic model $\mathcal{P}$ with a factor oracle. Let us consider that from state i, the only reachable states are state j and state 1. Using the context, μ_1, and μ_i, $\mathcal{P}$ is able to compute a score for the transition from state i to 1. Similarly, for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

Cluster graphs and message passing between oracles

In this section, we propose a model where 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 multi-agent systems that are more representative of a real free collective improvisation scenario. The method we propose could therefore be used to create a polyphonic and/or multidimensional improvisation, for instance a multi-instrument improvisation, a florid counterpoint or a melody / accompaniment duet. 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 their improvisation to have a real collective play. This could also represent the cognitive process of an individual musician 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 about what they are about to do to inform the others. This way every agent can make an informed decision accordingly. Message passing is organised on a graph representing which dimension each agent is working on and which dimensions it is listening to.

We first present the theoretical tools [Koller and Friedman (2009)] needed to use the belief propagation algorithm and then we present our method for multidimensional improvisation.

Cluster graph and message passing

Cluster graph

Let X be a set of random variables. A factor ϕ is a function from $\mathrm{Val}(X)$ to $\mathbb{R}$. The set of variables X is called the scope of the factor and noted $\mathrm{Scope}[\phi]$. Note that in our case, the notion of factor includes both joint probabilities and conditional probabilities. This will correspond to our sub-models.

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 a subset of variables $\mathcal{C}_i \subseteq X$ named cluster and each edge between two clusters $\mathcal{C}_i$ and $\mathcal{C}_j$ is associated with a sepset $\mathcal{S}_{i,j} \subseteq \mathcal{C}_i \cap \mathcal{C}_j$, that is a subset of variables shared by the two clusters about which they will communicate.

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

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

Figure 3 gives an example on how to distribute factors on a cluster graph. Note that, in this example, other distributions could have been chosen, for instance ϕ_2 could have been assigned to $\mathcal{C}_1$. In this case, we would have $\psi_1 = \phi_1.\phi_2$ and $\psi_2 = \phi_3$ instead.

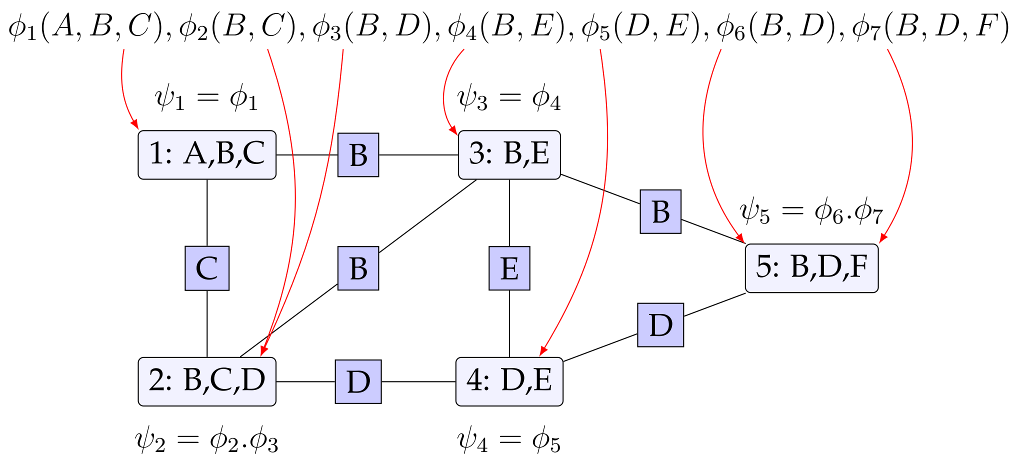

Figure 3. Example of factor distribution on a cluster graph.

A cluster graph must follow these properties (note that the example in Figure 3 satisfies them):

- *Family Preservation :* for each factor $\phi_k \in \Phi$, there must be a cluster $\mathcal{C}_i$ such as Scope$[\phi_k] \subseteq \mathcal{C}_i$. This way, we make sure 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.

- *Running Intersection Property :* for each pair $(\mathcal{C}_i, \mathcal{C}_j)$ of clusters and any variable $A \in \mathcal{C}_i \cap \mathcal{C}_j$, there is a unique path between $\mathcal{C}_i$ and $\mathcal{C}_j$ 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 including A. Second, the uniqueness of this path prevents the situation where the information about A goes in circle spawning false rumours.

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 noted $\delta_{i\to j}(\mathcal{S}_{i,j})$ and is defined by :

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

where N_i is the neighbourhood of i.

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

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

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

Note that $\delta_{i\to j}(\mathcal{S}_{i,j})$ does not depend on $\delta_{j\to i}(\mathcal{S}_{i,j})$. This prevents the repetition of the information we receive from a cluster 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 $\mathcal{C}_{\alpha(k)}$.

2. Compute the initial beliefs $\psi_i(\mathcal{C}_i) = \prod_{k:\alpha(k)=i} \phi_k$.

3. Initialise all the messages to 1.

4. Repeat message updates following formula (8).

5. Compute final beliefs :

$$\beta_i(\mathcal{C}_i) = \psi_i \prod_{k \in N_i} \delta_{k \to i} \tag{9}$$

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. $\beta_i(\mathcal{C}_i)$ is an approximation of the marginal probability $P(\mathcal{C}_i)$.

The convergence of the belief propagation algorithm is not guaranteed for any cluster graph. Note also that the order in which the messages are updated can have an influence on the convergence and on how fast it is. However, there is no way to determine the optimal order for message updates, this being completely dependent on the cluster graph construction. Generally, cyclic message updates have been proven to give the worst result in practice. In what follows, we have chosen to do message updates in a random order to avoid any bias.

The only particular case for which this algorithm converges every time towards an exact inference, for any order of message updates, is cluster trees. However, even if theoretical convergence is not guaranteed, this algorithm shows good results in practice [Koller and Friedman (2009)].

Communication between oracles for improvisation

Our goal is to use the combination of smoothed sub-models with the belief propagation algorithm on a cluster graph in order to make several factor oracles communicate with each other and therefore create a multidimensional improvisation where 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 by both the probabilistic modules defining the initial potentials and interpolation coefficients and the message passing between oracles through the cluster graph. This way, the oracles make a general choice of their path from internal and

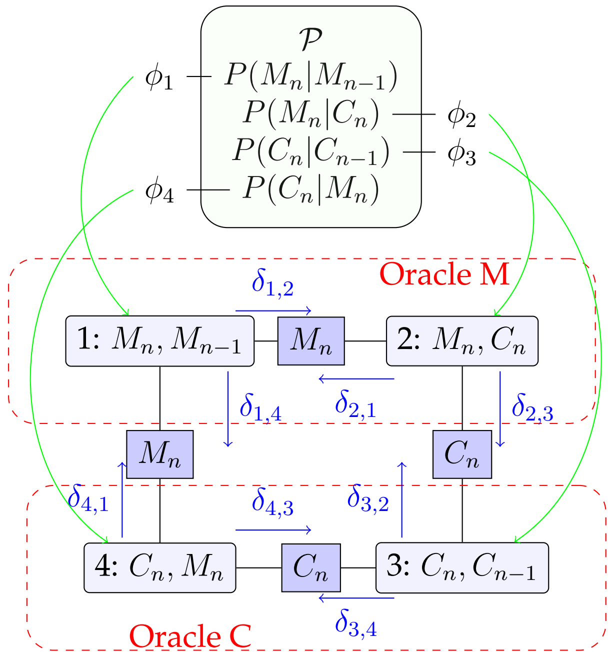

Figure 4. Cluster graph for multidimensional melody and harmony improvisation.

external knowledge.

In Figure 4 we show the cluster graph we used to create an improvisation with both melodic and harmonic data. We use n-gram models for melody and for harmony, respectively $P(M_n|M_{n-1})$ and $P(C_n|C_{n-1})$ and models representing the direct relations between melody and harmony : $P(M_n|C_n)$ and $P(C_n|M_n)$. Two oracles are constructed on the local context : Oracle M on melody and Oracle C on harmony. For each oracle two clusters are created : a first one for the temporal aspect of the dimension, and a second one for direct relation between the two dimensions. Note that this cluster graph respects both the family preservation and 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 module computes the factors ϕ_i corresponding to the smoothed sub-models. This provides the initial potential for each cluster of the graph. The messages $\delta_{i,j}$ for the belief propagation are then updated ten times in a random order. We then compute the final beliefs $\beta_i(\mathcal{C}_i)$ for each cluster. Therefore, we can, on the one hand, estimate $P(M_n)$ from $\beta_1(\mathcal{C}_1) = \beta(M_n, M_{n-1})$ or $\beta_2(\mathcal{C}_2) = \beta(M_n, C_n)$. For instance :

$$P(M_n) \simeq \sum_{C_n} \beta(M_n, C_n). \tag{10}$$

In theory, if the algorithm converged and produced exact inference :

$$\sum_{C_n} \beta(M_n, C_n) = \sum_{M_{n-1}} \beta(M_n, M_{n-1}) = P(M_n). \tag{11}$$

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

This model can be extended to a higher number of dimensions, musicians or a higher number of sub-models as long as the constructed cluster graph follows the family preservation and running intersection properties. Moreover, one of the main interest of this method is that it would be possible to use several probabilistic modules (one per oracle) trained on different corpora to emulate the style of different musicians, creating an individuality for each agent, and making this system more versatile than a system using a centralised knowledge with joint probabilities.

Experimentation

To evaluate the methods presented in this paper, we have generated improvisations using Charlie Parker's Omnibook [Parker and Aebersold (1978)] as a corpus. This corpus consists of 50 tunes composed, played and improvised on by Charlie Parker with symbolic melodic and harmonic data. This corpus can be found at [url hidden for submission anonimity]. This bebop jazz musician has a fairly distinctive style and is therefore a good choice to assess the style modelling of our methods. We divided this corpus into three sub-corpora : a training corpus consisting of 40 tunes and improvisations in order to train the different sub-models; a validation corpus consisting of 5 tunes and improvisations in order to optimise the interpolation and smoothing coefficients ; a test corpus consisting of 5 tunes and improvisations used to create the factor oracles during generation.

We conducted listening tests with Pascal Mabit, a professional jazzman, saxophonist and jazz teacher, graduated with a Master's Degree from the Conservatoire National Supérieur de Musique et de Danse de Paris in order to have a formal evaluation of the generated improvisation. As a jazz saxophonist, he is very familiar with the music of Charlie Parker and therefore able to provide us valuable feedback. Examples of generation for the different experiments are available at [url hidden for submission anonimity].

Factor oracle and probabilistic model

Guiding improvisation with a probabilistic model

In order to evaluate our model with a Factor Oracle exploiting a probabilistic model, we conducted two experiments. First, we generated free improvisations in order to compare the improvisation generated with a Factor Oracle alone and with a Factor Oracle combined with a probabilistic model. We chose to use two sub-models :

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

Those sub-models 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. However, the chord chart of the original tune is followed when generating an improvisation with the probabilistic module. We generated a dozen of improvisations by both methods on two tunes: Anthropology (a rhythm change), and Donna Lee.

Mabit noticed a clear difference between the two methods in terms of harmonic progression. Without the probabilistic model the harmony wasn't clear, or arranged in a random way (except when the improvisation was 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 sentences after sentences. And even inside sentences, there is not necessarily any harmonic sense."*

When using a probabilistic model, he was at some times able to say which chord the improvisation was playing on, despite it not being played. Moreover, he found that there was a clear sense of the succession of tonal centres. Despite that, the improvisation preserves the global style of Charlie Parker thanks to the local context provided by the Factor Oracle. On Donna Lee, Mabit also noticed that this method is not too constrained by the harmony and is able to play the whole range of a tonality.

> *"First of all, we can hear more of a chord progression, or rather of the tonal centre. All the beginning was in Bb Major, with turns of phrase, ornaments, things like that, that correspond to the be-bop or Charlie Parker's style.[...] We can really hear when it goes to the fourth degree, it plays the minor fourth degree, and then comes back..."*

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

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

This first experiment showed good results overall. The impact of probabilistic model can be noticed by a professional jazzman and the generated improvisations are preferred when using one. Some limitations of both systems were pointed out by Mabit, especially about the lack of global form.

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 offseting on the respective chord charts that are now being played along with the improvisation to highlight the melody/harmony relations. We first generated improvisations using the Omnibook corpus for training, and then using a training corpus consisting of about a thousand classical music tunes from all period instead. The Factor Oracles are in both cases constructed on Charlie Parker's tune.

At first, Mabit did not notice a big difference between the two corpora due to the dominance of the local context provided by the Factor Oracle. The elements of improvisations from Charlie Parker are strong and are the main focus, therefore the influence from classical music is less clear. But after a while, Mabit concluded 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 improvisation seemed more careful, and therefore sounded better from a harmony point of view.

"The most credible method in my opinion is the last one. 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. With just Charlie Parker, it follows the chord chart but it takes turns that don't always make sense."

At the end of this experiment, we can say that a difference can be noticed when using different corpora. However, unlike what we first thought, probabilistic training over a Charlie Parker corpus does not necessarily provide more realistic improvisation. This is mainly due to the fact that both methods share the same Factor Oracle providing a strong local context. Moreover, the classical music corpus provides harmonic information that is stricter and might be more realistic considering Charlie Parker's musical influences (Buster Smith, Lester Young, Stravinsky, and a lot of classical music).

Cluster graph and communication

To evaluate our interactivity model with 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.

Mabit thought that the generated improvisation were quite realistic, and could even represent a real life situation. The generated harmonic progression wasn't exactly the real chord chart but was logical and sounded like a real jazz song, and could have easily been played upon.

"It's funny, it really sounds like a wacky idea from the CNSM 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."

Mabit noticed that the melody followed the harmony properly, but might be too subordinated to the harmony, and therefore was less convinced by the generated melody that felt a bit bland at times and was not enough 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. [...] In improvisation, there is also a concept of reactivity, not just anticipation. There, it feels like there is only anticipation."*

Generally, the generated multidimensional improvisation seemed quite realistic and musical. Even if feeling a bit constrained by too much anticipation and a limited local context, this system improvises several dimensions with both a horizontal and vertical logic, and provides encouraging results.

Conclusion and discussion

We presented two methods able to learn multidimensional information in order to generate musical improvisations. First, we have shown the musical potentialities 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 can benefit from advanced smoothing techniques and optimisation for interpolation that make 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 closer way to a human improviser.

Second, we have introduced a method modelling 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 with a listening test conducted with a professional jazz saxophonist. Both methods received overall good feedback and seemed to be able to generate quite realistic improvisations. Some limitations of the current status of these methods were raised during the listening session, especially about the lack of 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 multi-scale organisation 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. in order to improve their results. This work also open the doors to musicology research to create more realistic avatars of musicians, trying to find out what were the influences of a musician and by training probabilistic models on a corpus comprising these influences, while focusing generation on the musician's own music through its factor oracle.

References

Allauzen, C., M. Crochemore, and M. Raffinot. 1999. "Factor oracle : a new structure for pattern matching." In *Proceedings of SOFSEM'99, Theory and Practice of Informatics*. 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., G. Bloch, and M. Chemillier. 2006. "OMax-OFon." In *Proceedings of the Sound and Music Computing Conference*.

Assayag, G., and S. Dubnov. 2004. "Using factor oracles for machine improvisation." *Soft Computing* 8-9:604–610.

Bellgard, M. I., and C. P. Tsang. 1999. "Harmonizing music the Boltzmann way." In N. Griffith, and P. M. Todd, (editors) *Musical Networks*. MIT Press, pp. 261–277.

Bickerman, G., S. Bosley, P. Swire, and R. M. Keller. 2010. "Learning to create jazz melodies using deep belief nets." In *Proceedings of the International Conference on Computational Creativity*. pp. 228–236.

Bimbot, F., G. Sargent, E. Deruty, C. Guichaoua, and E. Vincent. 2014. "Semiotic description of music structure: an introduction to the Quaero/Metiss structural annotations." In *Proceedings of the AES 53rd International Conference on Semantic*. pp. 32–43.

Chen, S. F., and J. Goodman. 1998. "An empirical study of smoothing techniques for language modeling." Technical Report TR-10-98, Harvard University.

Chen, S. F., L. Mangu, B. Ramabhadran, R. Sarikaya, and A. Sethy. 2009. "Scaling shrinkage-based language models." In *Proceedings of the 2009 IEEE Workshop on Automatic Speech Recognition & Understanding*. pp. 299–304.

Chomsky, N. 1996. *Studies on semantics in generative grammar*, volume 107. Walter de Gruyter.

Crispell, M. 2000. "Elements of improvisation." In J. Zorn, (editor) *Arcana : Musicians on Music*. pp. 190–192.

Donze, A., S. Libkind, S. A. Seshia, and D. Wessel. 2013. "Control improvisation with application to music." Technical Report UCB/EECS-2013-183, EECS Department, University of California, Berkeley.

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., G. Assayag, O. Lartillot, and G. Bejerano. 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." *University of Montreal, Department of Computer Science, CP* 6128.

Jelinek, F., and R. L. Mercer. 1980. "Interpolated estimation of Markov source parameters from sparse data." In *Pattern Recognition in Practice*. pp. 381–397.

Kalonaris, S. 2016. "Markov networks for free improvisers." In *Proceedings of the 42nd International Computer Music Conference*. pp. 181–185.

Klakow, D. 1998. "Log-linear interpolation of language models." In *Proceedings of the 5th International Conference on Spoken Language Processing*. pp. 1695–1698.

Koller, D., and N. Friedman. 2009. *Probabilistic graphical models : principles and techniques*. MIT Press.

Lefebvre, A., and T. Lecroq. 2000. "Computing repeated factors with a factor oracle." In *Proceedings of the 11th Australasian Workshop On Combinatorial Algorithms*. pp. 145–158.

Lerdahl, F., and R. Jackendoff. 1983. *A Generative Theory of Tonal Music*. MIT Press.

Nika, J., and M. Chemillier. 2012. "ImproteK, integrating harmonic controls into improvisation in the Filiation of OMax." In *Proceedings of the International Computer Music Conference*. pp. 180–187.

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 13th Sound and Music Computing Conference*. pp. 380–387.

Parker, C., and J. Aebersold. 1978. *Charlie Parker Omnibook*. Alfred Music Publishing.

Raczyński, S. A., S. Fukayama, and E. Vincent. 2013. "Melody harmonisation 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 2nd International Workshop on Musical Metacreation*. pp. 114–121.

Van Den Oord, A., S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. 2016. "Wavenet : a generative model for raw audio." *arXiv preprint* (arXiv:1609.03499).

Wang, C., and S. Dubnov. 2014. "Guided Music Synthesis with Variable Markov Oracle." pp. 55–62.