Experimenting with Algorithmic Composition Techniques

Jessica Kuo
jesskuo@stanford.edu

Horia Margarit
horia@stanford.edu

1 Introduction

Algorithmic composition, a form of artificial creativity, is not a new concept. Music is arguably the most mathematical art form in existence and since there has been music, composers have tried to develop processes to supersede the human creative process. Different structures were developed over time; notable examples include *i*) counterpoint in the Baroque era, which dictated strict rules for organized music writing, and *ii*) the ubiquitous sonata rondo form in the Classical era, which only supplied a high-level structure for an overall piece.

Pre-computer examples that could be considered algorithms include the famous *Musikalisches Wrfelspiel* ("musical dice game") implemented by W. A. Mozart, where the rolls of two six-sided dice randomly selected small sections of music that were then patched together to create a musical piece. This game was capable of producing $11^{16} = 45,949,729,863,572,161$ different yet similar waltzes [1]. Music composition algorithms have evolved into three main categories: *i*) aleatoric methods (e.g. Cage); *ii*) determinacy methods, where decisions over everything from notes to dynamic markings were objectified to pre-composed series and matrices of values (e.g. Schoenberg, Webern, and Berg); and *iii*) stochastic methods (e.g. Xenakis, Hiller). However, up to this point, the inputs still required manual creative input from the composer.

1.1 Motivation

In the computer age, researchers have employed Markov models and artificial neural networks to further reduce the level of manual input. The most notable software system was created by David Cope, and is called "Experiments in Musical Intelligence" (or "EMI") [2]. EMI uses a deconstruction method to first analyze existing pieces of music and separate it into parts and then second, to recombine it into a novel musical composition in the same style. This and other similar approaches have been criticized to only superficially manipulate or imitate the works of great composers and often do not take into the more "human" aspects of music, such as spontaneous rhythms, harmonies, timbre, or articulation. Further, most only apply to intra-genre mappings of musical sequences.

We intend to experiment with predicting the main key of a piece of music as well as the keys it may modulate to. In music, modulation is the act of changing from one key to another within a piece — it adds interest by shifting the tonal center. This may or may not be accompanied by a change in key signature (see Figure 1). The structures and rules of modulation vary greatly between genres and eras. With the exclusion of twelve-tone and atonal music, knowing the key a piece is written in is crucial for establishing tonality, which is the compositional foundation of all music. This should assist us in overcoming some of the issues outlined above and help us further reduce the imitative processes currently used in algorithmic compositions.

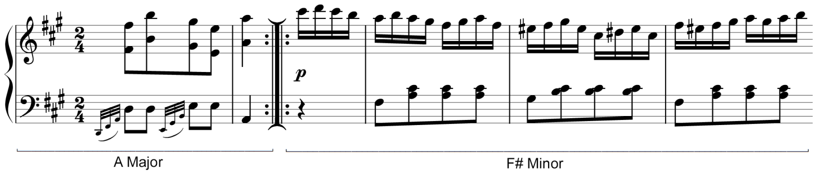


Figure 1: Excerpt from the third movement of Mozart's Piano Sonata No. 11, K. 331 showing modulation

2 Features

2.1 Dataset

MIDI files carry event messages that specify features such as notation, pitch, and dynamics in numerical values. This makes it easier for us to modify and manipulate music as data. Another advantage of MIDI is that it is compact and can be stored in a few kilobytes.

We selected 636 samples of music from the Baroque and Classical eras in MIDI format and manually loaded them into FL Studio, where we then checked them for accuracy and made edits wherever necessary. Each sample of music was then separated into its respective voices (e.g. 4 for keyboard fugues from the Baroque era, 4 for string quartets, 2 for certain keyboard pieces from the Classical era where separating the data further beyond left hand / right hand may impose sparsity, etc.). This was a time consuming process as the harmonic lines oftentimes came very close to or even crossed over one another. We ended up with a dataset of 1,465 samples.

2.2 Features

```
<meta message key_jk='A-'>
<meta message time_signature numerator=2 denominator=4 clocks_per_click=96
    notated_32nd_notes_per_beat=8 time=0>
<meta message set_tempo tempo=500000 time=0>
<meta message key_signature key='C' time=0>
note_on channel=0 note=71 velocity=59 time=1144
note_off channel=0 note=71 velocity=0 time=92
note_on channel=0 note=69 velocity=59 time=4
note_off channel=0 note=69 velocity=0 time=92
note_on channel=0 note=68 velocity=59 time=4
note_off channel=0 note=68 velocity=0 time=92
note_on channel=0 note=69 velocity=59 time=4
note_off channel=0 note=69 velocity=0 time=92
```

The above is an excerpt of features once again taken from the third movement of Mozart's Piano Sonata K. 331 after it had been imported and converted in Python. Since the key signature by definition does not distinguish between major and minor keys, we had to manually label the main key of each sample in our dataset for the purposes of error analysis.

The octave in which a certain musical note resides is irrelevant in evaluating the key and establishing tonality. Therefore, to eliminate any extraneous noise that may result from notes that are spread out, we will normalize the `note` data such that each musical note d is $d \equiv n \bmod 12$, where $n \in \{60, 61, \ldots, 71\}$. This is done so that our model will recognize each note, whether high or low, as one of the 12 notes from middle C to B. After that, we will take the n values and encode them as compressed sparse column matrices for input into our model.

The technical definition of our feature is outlined in section 2.2.2 .

2.2.1 Key

There are 12 major keys and 12 minor keys that music can be written in. The tonic note and chord of a piece of music gives the listener a subjective sense of arrival and completion.

2.2.2 Pitch

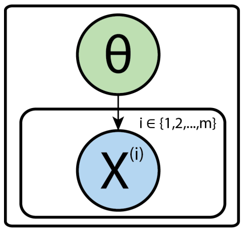
We represent the pitch of each musical note as a frequency data value d with respect to f, the frequency, where 440 Hz represents a concert A [3].

$$d = 69 + 12 \log_2 \left(\frac{f}{440\,\mathrm{Hz}} \right)$$

3 Methods

3.1 Model

We have constructed a generative model defined by the process and parameters below. The random variable X denotes the observed consecutive triplet of musical notes within any given sample with probability θ and where $X^{(i)}$ is some instantiation of X, encoded as a 1-of-K columnar vector of dimension K with exactly one element of value 1 and all other elements of value 0. Further details are included in section 3.2 .



$$\theta = (\theta_1, \ldots, \theta_K) \sim \text{Dirichlet}\,(\alpha) = \frac{1}{\text{Beta}(\alpha)} \prod_{j=1}^{K} \theta_j^{\alpha_j - 1} \propto \prod_{j=1}^{K} \theta_j^{\alpha_j - 1}$$

$$X^{(i)} \,|\, \theta = \left(x_1^{(i)}, \ldots, x_K^{(i)}\right) \sim \text{Categorical}\,(\theta) = \prod_{j=1}^{K} \theta_j^{x_j^{(i)}}$$

3.2 Deriving the Conjugate Prior

Let $X \sim$ Categorical(θ) with $x^{(i)}$ being some instantiation of the random variable, encoded as a 1-of-K vector such that it is columnar with dimension K, has exactly one element of the value 1 with all other elements of value 0. This allows us to write:

$$p(x^{(i)} | \theta) = \prod_{j=1}^{K} \theta_j^{x_j^{(i)}} = \exp\left(\log \prod_{j=1}^{K} \theta_j^{x_j^{(i)}}\right) = \exp\left(\sum_{j=1}^{K} \log \theta_j^{x_j^{(i)}}\right) = \exp\left(\sum_{j=1}^{K} x_j^{(i)} \log \theta_j\right)$$

This can be expressed in the general form of an exponential distribution: $h(x^{(i)}) \exp\left(\eta^T T(x^{(i)}) - A(\eta)\right)$ where $h(x^{(i)}) = 1$, $\eta^T = \log^T(\theta)$, $T(x^{(i)}) = x^{(i)}$, and $A(\eta) = 0$.

Now let $\theta \sim$ Dirichlet(α) with any instantiation of the random vector θ being a K-dimensional vector of reals, such that the j^{th} element yields the probability of the categorical variable X taking on value j. The sum of an instantiation of θ is therefore always 1.

$$\begin{aligned} p(\theta; \alpha) &= \frac{1}{\text{Beta}(\alpha)} \prod_{j=1}^{K} \theta_j^{\alpha_j - 1} \propto \prod_{j=1}^{K} \theta_j^{\alpha_j - 1} \\ &= \exp\left(\log \prod_{j=1}^{K} \theta_j^{\alpha_j - 1}\right) = \exp\left(\sum_{j=1}^{K} \log \theta_j^{\alpha_j - 1}\right) = \exp\left(\sum_{j=1}^{K} (\alpha_j - 1) \log \theta_j\right) \\ &= \exp\left(\sum_{j=1}^{K} \alpha_j \log \theta_j - \sum_{j=1}^{K} \log \theta_j\right) = \exp\left(\eta^T x\right) \end{aligned}$$

Here, $x = \alpha$ and $\eta^T = \log^T(\theta)$.

Now let us compute the joint probability using what we have derived so far.

$$\begin{aligned} p(x^{(1)}, \ldots, x^{(m)}, \theta\,; \alpha) &\propto \exp\left(\eta^T x\right) \prod_{i=1}^{m} \exp\left(\eta^T T(x^{(i)})\right) = \exp\left(\eta^T x\right) \exp \sum_{i=1}^{m} \left(\eta^T T(x^{(i)})\right) \\ &= \exp\left(\eta^T \left(x + \sum_{i=1}^{m} T(x^{(i)})\right)\right) = p\left(\theta\,;\, \alpha + \sum_{i=1}^{m} T(x^{(i)})\right) \\ &= p\left(\theta\,;\, \alpha + \sum_{i=1}^{m} x^{(i)}\right) \end{aligned}$$

But observe that by Bayes' Rule, the posterior $p(\theta \,|\, \ldots, x^{(i)}, \ldots \,;\, \alpha)$ is proportional to the joint probability we just calculated. They are in the same family and therefore, the prior and posterior are *conjugate distributions*, which makes the prior a *conjugate prior*. Note that "updating the prior" specifically amounts to the model updating its belief about the distribution of θ on every pass through of the data.

3.3 Parameter Estimates

We use the EM Algorithm on this model to compute the parameters of the latent variables.

3.3.1 E-Step

$$q^{(t)}(\theta \,|\, x^{(i)}) := \; p\left(\theta \,|\, x^{(i)};\, \alpha^{(t-1)}\right)$$

3.3.2 M-Step

$$\theta^{(t)} := \; \arg\max_{\theta} \; \sum_{i=1}^{m} q\left(\theta \,|\, x^{(i)}\right) \sum_{j=1}^{K} \left(x_j^{(i)} \log \theta_j + \alpha_j \log \theta_j\right)$$

4 Conclusion

One of the biggest hurdles in our modelling came into play when we realized that our original dataset lacked the crucial features for the learning algorithm we originally set out to build. The other big hurdle occurred when we attempted to model the parameter of the categorical distribution over X as being dependent on the central key of the musical piece. The Dirichlet conjugate prior over the parameter of the Categorical does not account for the hidden state — the central key — when it updates the frequency counts for the α parameter. This entails that our model and EM algorithm will converge on the maximum incomplete log likelihood that satisfies any central key. Intuitively, this maximum is less than the maximum that would have been obtained if it were conditioned on the central key.

Our current remedy to this problem is to use the fact that we arduously labelled the central keys of 636 musical pieces, which enables us to group the labelled musical pieces by central key. Then we could plate the graphical model (see section 3.1) such that we run one instance of the model on each group of musical pieces. The end result, and the goal of this entire endeavor, is to use EM and Bayesian inference to learn the optimal probability distribution of musical notes, given a particular central key of the musical score. To conclude, this endeavor provides us with the ability to run the model on data which is unlabeled and to select the central key under which the observed data is most likely.

5 Appendix

5.1 Deriving EM

$$\begin{aligned}
\log \prod_{i=1}^{m} p\left(x^{(i)}\right) &= \log \prod_{i=1}^{m} \int_{\theta} p\left(x^{(i)}, \theta \,;\, \alpha\right) \\
&= \log \left(\prod_{i=1}^{m} \int_{\theta} q(\theta \,|\, x^{(i)}) \frac{p\left(x^{(i)}, \theta \,;\, \alpha\right)}{q\left(\theta \,|\, x^{(i)}\right)} \right) \\
&= \sum_{i=1}^{m} \log \left(\int_{\theta} q(\theta \,|\, x^{(i)}) \frac{p\left(x^{(i)}, \theta \,;\, \alpha\right)}{q\left(\theta \,|\, x^{(i)}\right)} \right) \\
&= \sum_{i=1}^{m} \log \left(\mathbf{E}_{q(\theta \,|\, x^{(i)})} \left[\frac{p\left(x^{(i)}, \theta \,;\, \alpha\right)}{q\left(\theta \,|\, x^{(i)}\right)} \right] \right) \\
&\geq \sum_{i=1}^{m} \mathbf{E}_{q(\theta \,|\, x^{(i)})} \left[\log \frac{p\left(x^{(i)}, \theta \,;\, \alpha\right)}{q\left(\theta \,|\, x^{(i)}\right)} \right]
\end{aligned}$$

The inequality above results directly from Jensen's Inequality. Setting $q\left(\theta \,|\, x^{(i)}\right) := p\left(\theta \,|\, x^{(i)}\,; \alpha\right)$ results in equality [4]. We therefore have a lower bound on the *incomplete log likelihood* that is given by:

$$\sum_{i=1}^{m} \int_{\theta} q\left(\theta \,|\, x^{(i)}\right) \log \frac{p\left(x^{(i)}, \theta\,; \alpha\right)}{q\left(\theta \,|\, x^{(i)}\right)} \;\leq\; \log \prod_{i=1}^{m} p\big(x^{(i)}\big)$$

By recognizing that the denominator of the term inside the $\log$ is not a function of any of the parameters, as it is held constant during the M-Step, the overall expression to be maximized during the M-Step therefore reduces to:

$$\begin{aligned}\sum_{i=1}^{m} \int_{\theta} q\left(\theta \,|\, x^{(i)}\right) \log\left(p(x^{(i)} \,|\, \theta)\, p(\theta; \alpha)\right) &= \sum_{i=1}^{m} \int_{\theta} q\left(\theta \,|\, x^{(i)}\right) \log\left(\exp\left(\sum_{j=1}^{K} x_j^{(i)} \log\theta_j\right) \exp\left(\sum_{j=1}^{K} \alpha_j \log\theta_j\right)\right) \\ &= \sum_{i=1}^{m} \int_{\theta} q\left(\theta \,|\, x^{(i)}\right) \sum_{j=1}^{K}\left(x_j^{(i)} \log\theta_j + \alpha_j \log\theta_j\right)\end{aligned}$$

Since we will be taking the derivative with respect to θ, the integral cancels out and we end up having to maximize:

$$\sum_{i=1}^{m} q\left(\theta \,|\, x^{(i)}\right) \sum_{j=1}^{K}\left(x_j^{(i)} \log\theta_j + \alpha_j \log\theta_j\right)$$

References

[1] Zbikowski, Lawrence M. (2002). *Conceptualizing Music: Cognitive Structure, Theory, and Analysis*, pp. 142-143. New York: Oxford University Press.

[2] Cope, David (2006). *Computer Models of Musical Creativity*. Cambridge, MA: MIT Press.

[3] Midi Manufacturers Association, *Midi Tuning Specification*. Retrieved from `http://www.midi.org/techspecs/midituning.php`.

[4] Ng, Andrew (2015). *CS229 Lecture Notes: The EM Algorithm*. Retrieved from `http://cs229.stanford.edu/notes/cs229-notes8.pdf`.

[5] Marxer, Ricard & Purwins, Hendrik (2010). *Unsupervised Incremental Learning and Prediction of Music Signals*. Sydney, Australia: ISMA.

[6] Kosta, K., Marchini, M. & Purwins, H. (2012). *Unsupervised Chord-Sequence Generation from an Audio Sample*. Porto, Portugal: ISMIR.

[7] Marchini, Marco & Purwins, H. (2010). *Unsupervised Generation of Percussion Sound Sequences from a Sound Example*. In Sound and Music Computing Conference (Vol. 220).