Analysis and Synthesis of Palestrina-Style Counterpoint Using Markov Chains

Mary Farbood, Bernd Schoner*
MIT Media Laboratory
email: mary@media.mit.edu, schoner@media.mit.edu

Abstract

This paper presents a novel approach to computer-generated Palestrina-style counterpoint using probabilistic Markov Chains. It is shown how Markov Chains adequately capture the rules of species counterpoint and how they can be used to synthesize species counterpoint given a cantus firmus. It is also shown how such rules can be inferred from given counterpoint examples.

1 Introduction

Sixteenth-century counterpoint, with which the name of Palestrina has almost become synonymous, has long been held by musicians and theoreticians as an exceptionally elegant form of composition which has both musical and pedagogical value. Modeled after this style, species counterpoint was first introduced by J. J. Fux in his 1725 treatise, *Gradus ad Parnassum.* Fux codified the study of Palestrina-style counterpoint by presenting five categories of instruction, called species.

Gradus was a standard counterpoint text studied by countless musicians during the eighteenth and nineteenth centuries. However, from a stylistic viewpoint, it did not present an adequate approximation of Palestrina's music [SS89]. This goal was eventually realized by Knud Jeppesen, who believed that counterpoint must be studied with as much correspondence between written exercises and composition as possible [Jep31]. In other words, counterpoint had to be learned within the context of a specific musical style.

This paper presents a framework for generating music in the style of Palestrina using Markov models. We implement a set of species counterpoint rules, outlined by Jeppesen, in the form of probability tables which are then used as state-transition matrices for the Markov models. In the next step, the probability tables are estimated from given counterpoint examples.

The beauty and descriptive power of probabilistic network architectures and their graphical representation have been widely appreciated [Jor98]. In this paper we use Markov chains, a subclass of graphical models, which we believe are the appropriate tool for representing the compositional process of generating counterpoint. Markov chains provide the flexibility of integrating deterministic and probabilistic rules in the same model and can be trained on experimental data.

*Present address: ThingMagic LLC, Cambridge, MA 02142

2 Prior Work

In 1955, Hiller and Isaacson [HI58] began investigating techniques to generate music using the ILLIAC, a computer located at the University of Illinois. The first objective of their experiments was to "demonstrate that technical musical concepts could be translated into computer language to produce musical output." They used some basic counterpoint rules to generate cantus firmi, and then four-part first-species counterpoint. Ebcioglu created another rule-based system to generate fifth-species counterpoint given a cantus firmus which encapsulates each rule in a simple LISP function [Ebc80].

Lewin has implemented a program which generates first-species counterpoint using his own "Global Rule" in addition to the standard rules [Lew83]. It generates the counterpoint backwards from the cadence note in order to more easily implement the Global Rule. Although this method is more computationally efficient, it is, as argued by Robert Gjerdingen, an unmusical approach since it treats the counterpoint as a solution to a problem rather than an "aesthetic utterance."

Gjerdingen approaches the problem from a musician's point of view, as opposed to a programmer's [Gje88]. His system, PRAENESTE, is based on the idea that, just like a sixteenth-century singer improvising on a cantus, the computer should work forward in time without the benefit of being able to back up and start over. In response to each new contrapuntal situation, PRAENESTE uses a small number of concrete musical schemata to provide for itself a selection of correct melodic patterns.

Schottstaedt implemented all five species with up to six voices [Sch89]. His program follows Fux's guidelines and rules as closely as possible. These rules were extended and modified until the results acceptably resembled Fux's examples. Since Fux stresses that most of the rules are guidelines and not absolute, a penalty system is used to assign them relative degrees of importance. If at any given time the total penalty exceeds a certain amount, that particular path is abandoned.

Our approach to computer-generated counterpoint is considerably different from the cited work, because we (a) use a probabilistic description of the problem and (b) we infer our model from data which ensures that the result is musical assuming that the training examples themselves are musical.

3 Synthesizing Counterpoint

3.1 Dynamic Programming

We use a forward dynamic programming approach to find the most likely solution out of a multitude of possible solutions given a cantus firmus [Ber95].

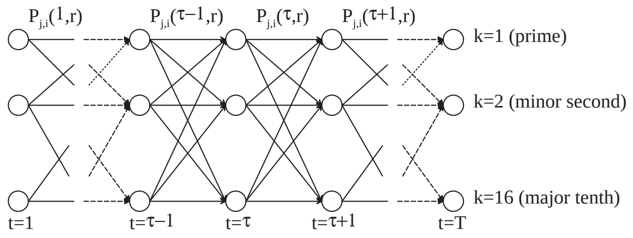

Figure 1: Trellis state diagram. The vertical states n correspond to note intervals relative to the cantus note at time t. The time t corresponds to time steps going forward. $P_{j,i}(t,r)$ denotes the probability of transition from state i at time t to state j at time $t+1$ with respect to rule r.

We start with the construction of a state trellis diagram where the states describe the possible chromatic notes in the counterpoint (fig. 2). We constrain the counterpoint to be above or below the cantus and allow a note range which spans a major tenth above (or below) the highest (or lowest) cantus note. Hence the number of states in the trellis equals the chromatic range of the cantus plus 17. We then formalize the musical constraints on the counterpoint in the form of a mix of transition probabilities and state probabilities conditioned on the cantus. For example, we specify the unconditional likelihood of a harmonic interval, or the probability of a melodic interval given the previous melodic interval.

While most rules can be realized based on a first-order Markov chain, some require a second-order system. We implement a basic first-order structure, but allow for second-order considerations by looking one step ahead and choosing optimal solutions with respect to the next time step.

In the synthesis application, we seek to find the most likely path through the trellis structure given the underlying cantus and transition rules. Evaluating every possible path would be computationally prohibitive. However, we make the problem tractable by using a dynamic programming approach commonly known as the Viterbi algorithm, which searches through the tree iteratively. The Viterbi algorithm is based on the insight that the most likely past state sequence leading into any one state is independent from future states.

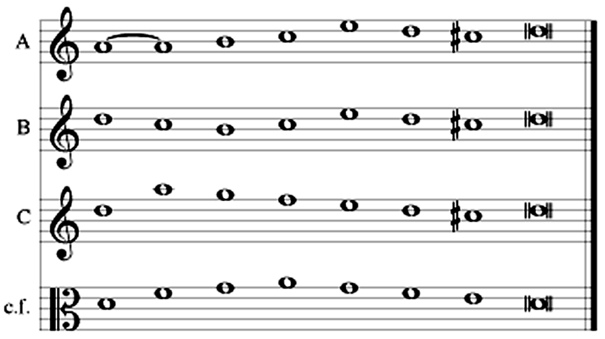

Figure 2: Counterpoint examples for a given cantus firmus. Example A was composed by a human (David Lewin). Example B was machine generated retaining the same climax point. Example C was machine generated with a different climax point.

Going forward in the chain, we multiply the previous path probabilities with the transition probability of the new update, and keep the path that generated the highest joint probability. At any time step τ we only consider the path leading into state $\tau-1$, the probability associated with that sequence, and the transition probabilities at time τ. After the last transition occurs, we select the sequence with the highest probability as the solution [Ber95].

3.2 Probabilistic Counterpoint Rules

Each rule is implemented as a probability table where illegal transitions are described by probability zero. The transition probabilities for generating a counterpoint line are obtained by multiplying the individual values from each table, assuming the rules are independent:

- The **harmonic table** determines whether the interval between the counterpoint note and its respective cantus note at any time τ is permissible. The likely intervals—third, sixth, tenth—are assigned higher probability values, whereas the forbidden intervals are assigned zero (table 1). For the first cantus note, only prefect intervals are permitted, and for the last, only unisons and octaves.
- The **melodic table** describes the likelihood of a melodic interval created by two consecutive counterpoint notes being followed by any other melodic interval (table 2).

Table 1: Probability of harmonic intervals between cantus and counterpoint for notes other than the first and last. *Top row:* Estimated from human-generated counterpoint (12 examples). *Middle row:* Edited probability table used in the generating program. *Bottom row:* Estimated from machine-generated counterpoint (44 examples).

	P1	m2	M2	m3	M3	P4	d5	P5	m6	M6	m7	M7	P8	m9	M9	m10	M10
Human counterp.	0	0	0	0.133	0.133	0	0	0.060	0.181	0.349	0	0	0.072	0	0	0.145	0.072
Edited table	0	0	0	.1428	.1428	0	0	.1428	.1428	.1428	0	0	.0004	0	0	.1428	.1428
Machine counterp.	0	0	0	0.100	0.104	0	0	0.054	0.118	0.409	0	0	0.133	0	0	0.172	0.065

Table 2: Melodic transition rules. The first column indicates the previous melodic interval in the counterpoint line. The top row indicates the next melodic interval. The entries indicate the probability of each transition. The top table shows the probabilities used in the generating program. The bottom table shows probabilities estimated from the machine generated counterpoint (44 examples).

Table edited for use in the generating program

	m2u	M2u	m3u	M3u	P4u	P5u	m6U	P8u	m2d	M2d	m3d	M3d	P4d	P5d	P8d	P1
m2u	0	.45	.2	.2	0	0	0	0	.035	.035	.025	.025	.01	.01	.009	.001
M2u	.45	.45	.03	.03	0	0	0	0	.01	.01	.005	.005	.004	.004	.002	.001
m3u	.45	.45	.03	.03	0	0	0	0	.01	.01	.005	.005	.004	.004	.002	.001
M3u	.35	.35	.05	.05	0	0	0	0	.05	.05	.025	.025	.025	.025	.024	.001
P4u	.065	.065	0	0	0	0	0	0	.4	.4	.02	.02	.01	.01	.009	.001
P5u	0	0	0	0	0	0	0	0	.52	.52	.01	.01	.01	.01	.009	.001
m6u	0	0	0	0	0	0	0	0	.51	.51	.025	.025	.01	.01	.009	.001
P8u	0	0	0	0	0	0	0	0	.51	.51	.025	.025	.01	.01	.009	.001
m2d	.05	.05	.005	.005	.002	.002	.002	.002	0	.467	.2	.2	.015	0	0	.001
M2d	.05	.05	.005	.005	.002	.002	.002	.002	.367	.367	.1	.1	.015	0	0	.001
m3d	.366	.366	.005	.005	.002	.002	.002	.002	.05	.05	.1	.1	0	0	0	.001
M3d	.366	.366	.005	.005	.002	.002	.002	.002	.05	.05	.1	.1	0	0	0	.001
P4d	.53	.53	.02	.02	.02	.02	.039	.02	0	0	0	0	0	0	0	.001
P5d	.454	.454	.03	.03	.01	.005	.005	.011	0	0	0	0	0	0	0	.001
P8d	.454	.454	.03	.03	.01	.005	.005	.011	0	0	0	0	0	0	0	.001
P1	.1	.1	.08	.08	.05	.05	.04	.03	.1	.1	.08	.08	.05	.04	.02	0

Table estimated from computer generated examples

	m2u	M2u	m3u	M3u	P4u	P5u	m6U	P8u	m2d	M2d	m3d	M3d	P4d	P5d	P8d	P1
m2u	0	.280	0	.080	0	0	0	0	.440	0	.160	0	0	0	0	.040
M2u	.088	.235	.029	0	0	0	0	0	0	.412	.029	.059	.029	.059	0	.059
m3u	0	.227	0	.045	0	0	0	0	.682	0	0	0	0	0	0	.045
M3u	.154	0	.077	0	0	0	0	0	0	.538	0	.077	.077	.077	0	0
P4u	.048	.238	0	0	0	0	0	0	.333	.286	.095	0	0	0	0	0
P5u	0	0	0	0	0	0	0	0	0	.833	0	.167	0	0	0	0
m6u	0	0	0	0	0	0	0	0	.500	0	.250	0	0	.250	0	0
P8u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
m2d	.581	0	.054	0	.027	0	.014	0	0	.189	0	.041	.041	0	0	.054
M2d	0	.070	0	.085	.099	0	.028	0	.423	.169	.113	0	0	0	0	.014
m3d	.235	.353	.176	0	0	.059	0	0	.059	.118	0	0	0	0	0	0
M3d	0	.143	0	0	.143	0	0	0	.429	0	.143	0	0	0	0	.143
P4d	0	0	.875	0	0	.125	0	0	0	0	0	0	0	0	0	0
P5d	0	.200	.200	.400	.200	0	0	0	0	0	0	0	0	0	0	0
P8d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P1	.133	.133	.067	.067	.200	.067	0	0	.200	.133	0	0	0	0	0	0

This table not only implements the rules determining legal melodic intervals and melodic patterns, but also has a profound effect on the shape of the line as a whole, since stepwise motion (among other such desirable behavior) is heavily weighted.

- The **cadence table**, in conjunction with the chromatic table, ensures that there is an appropriate cadence at the end of each example. Only stepwise motion is permitted to the final note, and stepwise motion to the penultimate note is given very high probability.
- The **chromatic table** gives very low weight to chromatically-altered notes (except the penultimate)—low enough that they will not be used unless there are no other solutions. For the penultimate note, only chromatically-altered notes, with the exception of Eb, are permitted, in order to provide a leading tone (the E-based Phyrgian mode does not have a raised leading tone).
- The **good parallel motion table** prevents too many consecutive parallel thirds, sixths, and tenths. The probability that three in a row can occur is small, and the probability that four or more can occur is minute.
- The **approaching motion table** prohibits approaching a perfect interval in direct motion (also known as "hidden" fifths or octaves).
- The **departing motion table:** prohibits leaving a unison in direct motion. This also has the added advantage of preventing the overlap of voices.
- The **general motion table** assigns contrary motion high probability, oblique and direct motion low probability.
- The **leap and climax tables** return probabilities of zero or one. The leap table returns zero if both the cantus and the counterpoint are leaping simultaneously in the same direction and have done so previously (only one such occurrence is permitted). The climax table determines whether a pitch is acceptable given restrictions set by the chosen climax position of the counterpoint.

4 Analyzing Counterpoint

In this section we show how the probabilistic description of counterpoint rules can be inferred from existing compositions. We start by composing a set of first-species counterpoint strictly conforming to the rules. Twelve examples were composed by humans, 44 examples were generated by a machine using the algorithm described above. This database of species-counterpoint examples is used as training data for a Markov model. We set up the framework for a state sequence identical to the last section. We then infer the transition tables by counting the number of occurrences of certain transitions and renormalizing. Figures 1 and 2 compare tables which were used in the generating algorithm with those that were inferred from human-generated examples and machine-generated examples.

5 Experimental Results

A software application for Windows was written that (a) infers probability tables from given counterpoint examples, (b) composes first-species counterpoint given a cantus firmus, (c) plays and displays the composed music.

Much of the experimentation centered around deciding appropriate transition probabilities for the rule tables. While it was easy to make transitions legal or illegal, based on the outlined rules, it was more interesting and time-consuming to weight the probabilities properly in order to get musical results. Subtle changes in the table values resulted in very different solutions. Also probabilities had to be weighted not just with respect to other values in the same table but with respect to competing rules. For example, one possible situation where there are two acceptable solutions might be the following: (1) the penultimate note is approached in stepwise motion (ideal) but four consecutive parallel sixths result (acceptable, but not ideal) vs. (2) the penultimate note is not approached in stepwise motion (acceptable, but not ideal) but there are fewer consecutive parallel sixths. This is just one example of an aesthetic decision which can be reflected in the probability values.

During the course of implementation, it became apparent that some of the earlier tables were unnecessary. For example, there was initially a table which restricted parallel fifths and octaves. This was later rendered superfluous by the *approaching motion* table listed above (parallel motion is a special case of direct motion).

Another influential aesthetic factor was climax placement. As recommended by Jeppesen, the climax note should be unique as well as higher in pitch than all other notes. The program either tries each possible climax time until a satisfactory (if any) solution is reached, or the user specifies where the climax point should be. This rule helps produce surprisingly musical results.

6 Conclusions and Future Work

It was shown that a Markov structure can be used to compose a first-species counterpoint line given a cantus firmus. Not only does the composed line comply with the strict rules of sixteenth-century counterpoint, but the results are also musical and comparable to those created by a knowledgeable musician. The program was capable of finding solutions identical to those presented by Jeppesen as well as some of the student compositions by one of the authors ... graded *A*. It was furthermore shown how some of the more complex transition tables can be estimated from given counterpoint examples. It was found that the probability tables are close to the data available for estimation.

The ultimate goal of this work is to infer rules from authentic Palestrina works and to compose such counterpoint based on the inference model. The authors have collected a MIDI database consisting of 30 three-part movements from Palestrina masses, yet the evaluation of the data must be left for future work.

7 Acknowledgements

The authors would like to thank Tod Machover and Neil Gershenfeld for their support. This work was made possible by the MIT Media Lab's Things That Think consortium and by the Sega Corporation.

References

[Ber95] Dimitri Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont Massachusetts, 1995.

[Ebc80] Kemal Ebcioglu. Computer counterpoint. In *Proceedings International Computer Music Conference*, New York, 1980.

[Gje88] Robert Gjerdingen. Concrete musical knowledge and a computer program for species counterpoint. In *Explorations in Music, the Arts, and Ideas: Essays in Honor of Leonard B. Meyer.* Pendragon Press, Stuyvesant, 1988.

[HI58] Lejaren Hiller and Leonard Isaacson. Musical composition with a high-speed digital computer. *Journal of the Audio Engineering Society*, 1958.

[Jep31] Knud Jeppesen. *Counterpoint, the Polyphonic Vocal Style of the Sixteenth Century.* Dover Publications, New York, 1931.

[Jor98] Michael Jordan, editor. *Learning in Graphical Models.* MIT Press, Cambridge, Massachusetts, 1998.

[Lew83] David Lewin. An Interesting Global Rule for Species Counterpoint. *Theory Only*, 6(8):19–44, 1983.

[Sch89] William Schottstaedt. Automatic counterpoint. In Matthews and Pierce, editors, *Current Directions in Computer Music Research.* MIT Press, 1989.

[SS89] Felix Salzer and Carl Schachter. *Counterpoint in Composition.* Columbia University Press, New York, 1989.