User Experience and Efficacy of the Exp4 Algorithm for Algorithmic Composition

Robert Harlow
4399 County Road T
Barneveld, WI 53507
rharlow@wisc.edu

Abstract

I use the Exp4 Algorithm to learn to algorithmically compose music that appeals to the taste of a human oracle. In each round, experts based on “major composers" give advice on whether to recommend a ten second snippet of music that has 1 - 7 distinct pitch classes, 8 or 9 distinct pitch class, or 10 - 12 distinct pitch classes. I derive bounds on the expected regret of Exp4 in this setting, I implement a web application were the Exp4 Algorithm can be run, and I discuss user experience and the utility of implementing Exp4 in this way. I find that by comparing my independent ranking of how well I like the composers used for the experts with the rankings that Exp4 produced over three sessions totaling slightly less than an hour in duration, the hypothesis that the rankings produced by Exp4 are random has less than a 10% chance of producing the observed rankings.

1 Introduction

“Algorithmic composition" is the automatic generation of new music using a computer. A large body of research on algorithmic composition exists, encompassing the use of LSTM neural networks, Markov models, grammars, and evolutionary algorithms. To the best of my knowledge, this paper is the first use of a bandit algorithm for algorithmic composition.

I use the Exp4 Algorithm [12] to learn to algorithmically compose music that appeals to the taste of a human oracle. In each round, experts based on “major composers" give advice on whether to recommend a ten second snippet of music that has 1 - 7 distinct pitch classes, 8 or 9 distinct pitch class, or 10 - 12 distinct pitch classes based on the preferences found in those composers’ work.

I derive bounds on the expected regret of Exp4 in this setting and implement a web application were the Exp4 Algorithm can be run.

I discuss user experience and the utility of implementing Exp4 in this way.

I find that by comparing my independent ranking of how well I like the composers used for the experts with the rankings that Exp4 produced over three sessions totaling slightly less than an hour in duration, the hypothesis that the rankings produced by Exp4 are random has less than a 10% chance of producing the observed rankings.

I also note that it is rare to be able to “experience" a learning algorithm in an intuitive setting, as one perhaps might in other fields, in an introductory Chemistry lab or on a Geology field trip. I hope that users will enjoy interacting with the Exp4 Algorithm through the web application that I developed, and that the web application will give users increased intuition for the performance of Exp4.

Preprint. Under review.

2 Related work

A large body of research on algorithmic composition exists, encompassing the use of LSTM neural networks, Markov models, grammars, and evolutionary algorithms.

2.1 LSTM neural networks

- Choi et al. [9] implement two case studies on chord progressions and drum tracks respectively. In both, the music is abstracted to sequences in a text document and LSTM neural networks are trained on the sequences and subsequently used to generate new sequences.
- Liu and Ramakrishnan [13] train a LSTM neural network on Bach chorales and show that it can reproduce the chorales it is trained on by feeding the LSTM neural network the first notes of each chorale in the training set. They use resilient propagation (RPROP) as an alternative to standard back-propagation.

2.2 Markov models

- Farbood and Schöner [10] train a Markov model to compose using counterpoint. They subsequently generate new music by using their model to compose an accompanying part against a fixed melody.

2.3 Grammars

- Keller and Morrison [11] use probabilistic grammars to computationally generate novel Jazz improvisation over fixed chord progressions.

2.4 Evolutionary algorithms

- Moroni et al. [14] use evolutionary algorithms to computationally generate new music that best meets fitness criteria selected by a human.

3 Overview of my approach

I define three actions: 1.) Play a ten second snippet of music that has up to seven distinct pitch classes (a pitch's "pitch class" is the set of all pitches that are a whole number of octaves away). 2.) Play a ten second snippet of music that has either 8 or 9 distinct pitch classes. 3.) Play a ten second snippet of music that has 10 or more distinct pitch classes.

Experts based on major composers advise what action to take. Their advice is based on the frequency that those composers employ the actions in selected works, evaluated by splitting midi files representing those works into 10 second intervals ($[0s, 10s), [10s, 20s) \dots$), and for each interval, counting the number of distinct pitch classes used. Table 1 summarizes the advice of the experts.

The "user" of the algorithm gives feedback by choosing between zero and five "stars" of reward for each snippet of music. 0, 1, 2, 3, 4, and 5 stars of reward are respectively equivalent to rewards of 0, 0.2, 0.4, 0.6, 0.8, and 1 in pure numbers.

The Exp4 Algorithm is employed to adapt to the user's feedback.

4 Regret bound

In section 18.4 of Lattimore and Szepesvári [12], it is shown that for the Exp4 Algorithm, if the learning rate, η, is set to $\eta := \sqrt{\frac{2\log(M)}{nk}}$ where M is the number of experts, k is the number of actions, and n is the number of rounds, and if the parameter γ is set to $\gamma := 0$, then, letting R_n denote the expected regret of Exp4 after n rounds, the Exp4 Algorithm has regret bound $R_n \leq \sqrt{2nk\log(M)}$.

In my approach η, is set to $\eta := \sqrt{\frac{2\log(M)}{nk}}$ and γ is set to $\gamma := 0$. There are three actions, so $k = 3$. The number of experts M ranges between two and twelve depending on user preference. The number of rounds n is selected by the user. Table 2 lists representative regret bounds as a function of M

Table 1: The advice that the experts give. The numbers denote the probability rounded to three significant figures that an expert recommends using the associated range of distinct pitch classes for a ten second snippet of music.

	Number of distinct pitch classes		
Expert	1 - 7	8 or 9	10 - 12
Leonard Bernstein	0.106	0.202	0.692
Claude Debussy	0.25	0.198	0.552
Antonín Dvořák	0.397	0.407	0.196
Gustav Holst	0.322	0.299	0.378
Gustav Mahler	0.264	0.338	0.398
Wolfgang Amadeus Mozart	0.0972	0.375	0.528
Sergei Prokofiev	0.02	0.08	0.9
Nikolai Rimsky-Korsakov	0.293	0.308	0.399
Dmitri Shostakovich	0.172	0.255	0.573
Jean Sibelius	0.292	0.323	0.385
Igor Stravinsky	0.149	0.159	0.692
Richard Wagner	0.166	0.29	0.545

Table 2: The expected regret of Exp4 after n rounds as a function of the number of experts, M, and the number of rounds, n, for representative values of M and n.

	n									
M	20	40	60	80	100	120	140	160	180	200
2	9.12	12.9	15.8	18.2	20.4	22.3	24.1	25.8	27.4	28.8
3	11.5	16.2	19.9	23	25.7	28.1	30.4	32.5	34.4	36.3
4	12.9	18.2	22.3	25.8	28.8	31.6	34.1	36.5	38.7	40.8
5	13.9	19.7	24.1	27.8	31.1	34	36.8	39.3	41.7	43.9
6	14.7	20.7	25.4	29.3	32.8	35.9	38.8	41.5	44	46.4
7	15.3	21.6	26.5	30.6	34.2	37.4	40.4	43.2	45.8	48.3
8	15.8	22.3	27.4	31.6	35.3	38.7	41.8	44.7	47.4	50
9	16.2	23	28.1	32.5	36.3	39.8	43	45.9	48.7	51.3
10	16.6	23.5	28.8	33.2	37.2	40.7	44	47	49.9	52.6
11	17	24	29.4	33.9	37.9	41.6	44.9	48	50.9	53.6
12	17.3	24.4	29.9	34.5	38.6	42.3	45.7	48.8	51.8	54.6

and n. Table 3 lists representative “average expected regret per round in units of stars" bounds as a function of M and n. The representative bounds are rounded to three significant figures.

It is important that a good regret bound can be achieved in a “tolerable" number of rounds from the perspective of a human user. Based on Table 2 and Table 3, I believe my approach succeeds for that criteria.

5 Preparing the data

Midi files of works of music were sourced from

http://www.scena.org/midi/music/index.html [15]

The midi file selected are partial and/or complete representations of *West Side Story* by Leonard Bernstein, *La mer* by Claude Debussy, Antonín Dvořák's Ninth Symphony, *The Planets* by Gustav Holst, Gustav Mahler's Fifth Symphony, Wolfgang Amadeus Mozart's 41st Symphony, Sergei Prokofiev's Fifth Symphony, *Scheherazade* by Nikolai Rimsky-Korsakov, *Festive Overture* by Dmitri Shostakovich, Dmitri Shostakovich's Tenth Symphony, Jean Sibelius's Second Symphony, *Petrushka* and *The Rite of Spring* by Igor Stravinsky, and *Tristan und Isolde* and *Tannhäuser* by Richard Wagner.

Table 3: The "average expected regret per round in units of stars" of Exp4 after n rounds as a function of the number of experts, M, and the number of rounds, n, for representative values of M and n.

	n									
M	20	40	60	80	100	120	140	160	180	200
2	2.28	1.61	1.32	1.14	1.02	0.931	0.862	0.806	0.76	0.721
3	2.87	2.03	1.66	1.44	1.28	1.17	1.08	1.01	0.957	0.908
4	3.22	2.28	1.86	1.61	1.44	1.32	1.22	1.14	1.07	1.02
5	3.47	2.46	2.01	1.74	1.55	1.42	1.31	1.23	1.16	1.1
6	3.67	2.59	2.12	1.83	1.64	1.5	1.39	1.3	1.22	1.16
7	3.82	2.7	2.21	1.91	1.71	1.56	1.44	1.35	1.27	1.21
8	3.95	2.79	2.28	1.97	1.77	1.61	1.49	1.4	1.32	1.25
9	4.06	2.87	2.34	2.03	1.82	1.66	1.53	1.44	1.35	1.28
10	4.16	2.94	2.4	2.08	1.86	1.7	1.57	1.47	1.39	1.31
11	4.24	3	2.45	2.12	1.9	1.73	1.6	1.5	1.41	1.34
12	4.32	3.05	2.49	2.16	1.93	1.76	1.63	1.53	1.44	1.37

Each ten second "snippet" of each midi file ($[0s, 10s), [10s, 20s) \dots$) was analyzed with the "mido"[3] Python library to determine the number of distinct pitch classes present. Each ten second snippet of each midi file was "recorded" in mp3 format using the open source libraries "FluidSynth"[2] and "FFmpeg"[1].

When an action is chosen, one of the snippets of music consistent with the action is choose at random to recommend to the user. For example, if the action is "play a ten second snippet of music that has up to seven distinct pitch classes", then a snippet of music is chosen at random with uniform probability from the set of snippets that meet that criteria.

6 Building the web application

A web application implementing "Exp4" as presented in this paper was built using Python, HTML, and Amazon Web Services (AWS) [4]. The web application's backend is built with AWS Lambda [5], Amazon DynamoDB [7], Amazon API Gateway [6], and Amazon S3 [8].

7 GitHub repository

The GitHub repository containing my code for this project is:

https://github.com/robertwharlow/exp4_music

8 Using the web application

First, the audio files in the web application are not normalized for volume, and some are quite soft or loud, so take care to keep the volume low and increase if needed.

The web application can be accessed at

https://wlzev4j6c2.execute-api.us-east-1.amazonaws.com/{user}

where "{user}" is a placeholder for a URL "path parameter" that is used to separate the sessions of distinct users. Any value for "{user}" is valid. For example, a user named "Kaitlin" might choose https://wlzev4j6c2.execute-api.us-east-1.amazonaws.com/kaitlin.

When a user first visits the web application, a new session is started by default with the number of experts $M := 3$ and the number of rounds $n := 30$. A new session can be started at any time with user selected values for M and n using an HTML form on the page.

To run the Exp4 Algorithm, the user repeatedly selects the number of stars of reward they wish to assign to a snippet of music and submits that reward using the "Submit reward" button.

When the end of the n rounds is reached, and the Exp4 Algorithm thus completes, the user is presented with the final weights assigned to the various experts as well as an option to start a new “session" of Exp4 with user selected values for M and n.

9 User experience

User experience is an important consideration in implementing an interactive learning algorithm. Important questions are, Is it too boring and repetitive for users? Is the regret bound that can practically be achieved relevant?

I participated in three sessions of Exp4. For the first session, the number of experts M was set to 2 and the number of rounds n was set to 40. For the second session, $M := 12$ and $n := 100$. For the third session, $M := 3$ and $n := 45$. The sessions took me 13 minutes, 29 minutes, and 12 minutes to complete respectively.

I did not find the first or third sessions boring, but the second session did get slightly boring. I was taking notes, and during round 60, I jotted on my note paper that I was “bored". For the shorter sessions, the large variety of “snippets" of music that could be played (there are 2,524 snippets by 12 different composers), my curiosity for how good the midi would sound compared to live instruments, and the fact that I am not well versed on some of the works in the “library" kept my interest.

For the three sessions I participated in, the “average expected regret per round in units of stars" were 1.61, 1.93, and 1.91 stars respectively. Since many of the rewards I submitted were either 0 or 5 stars, a guarantee that on average, the recommended “snippet" is less than 2 stars away from the optimal snippet does seem meaningful to me. If the number of rounds was tripped for each session, the “average expected regret per round in units of stars" would instead be 0.93, 1.11, and 1.10 stars respectively, which seems even more worthwhile to me. Although I would almost surely suffer some boredom with the increased duration of the sessions, the number of rounds is still practical.

10 Efficacy

If Exp4 works well, then it should be good at learning my musical preferences. To test how well Exp4 did, I logged three of my sessions. The evolution of the weights for the experts for the three sessions is plotted in Figures 1, 2 and 3 respectively. Additionally, I ranked the “experts" based of my prior knowledge of how well I like the various composers. My ranking (independent of Exp4) is:

1. Sergei Prokofiev
2. Dmitri Shostakovich
3. Jean Sibelius
4. Igor Stravinsky
5. Leonard Bernstein
6. Nikolai Rimsky-Korsakov
7. Richard Wagner
8. Claude Debussy
9. Gustav Mahler
10. Gustav Holst
11. Antonín Dvořák
12. Wolfgang Amadeus Mozart

To gauge how well Exp4 learned my preferences, I computed the length of the longest increasing subsequence between my ranking and the final rankings produced by the Exp4 sessions. I also computed the empirical likelihood that a random ranking and my ranking would have a longest increasing subsequence of greater or equal length by evaluating 10,000 random rankings. The lengths of the longest increasing subsequences for Sessions 1, 2 and 3 are 2, 6, and 2 respectively, and the empirical probabilities that a random ranking would have a greater than or equal length are 0.4985, 0.2157, 0.8399 respectively. It follows that if the rankings produced by Exp4 are random, than

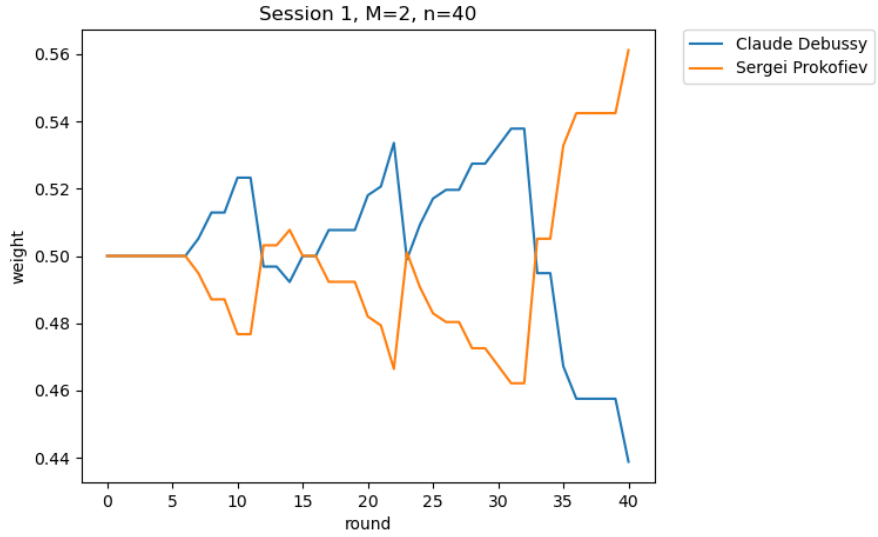

Figure 1: Evolution of the weights for the experts during my first session of Exp4 with the number of experts $M := 2$ and the number of rounds $n := 40$.

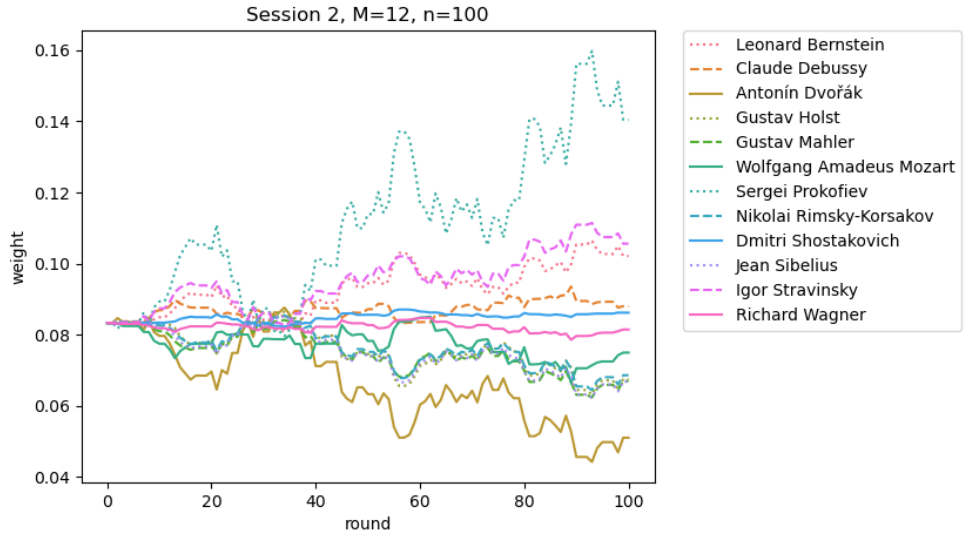

Figure 2: Evolution of the weights for the experts during my second session of Exp4 with the number of experts $M := 12$ and the number of rounds $n := 100$.

the empirical likelihood of observing three rankings that yield longest increasing subsequences of greater than or equal length compared with those yielded by the three ranking that Exp4 produced is $0.4985 \times 0.2157 \times 0.8399 = 0.0903$. Although not statistically significant, this result does lend some weight to the hypothesis that Exp4 was successful at learning my preferences.

11 Future work

The actions investigated in the paper (the number of distinct pitch classes present in a 10 second snippet of music) are convenient because they neatly partition any set of “snippets" of music into a small number of disjoint subsets, and also because “major composer" experts can naturally recommend

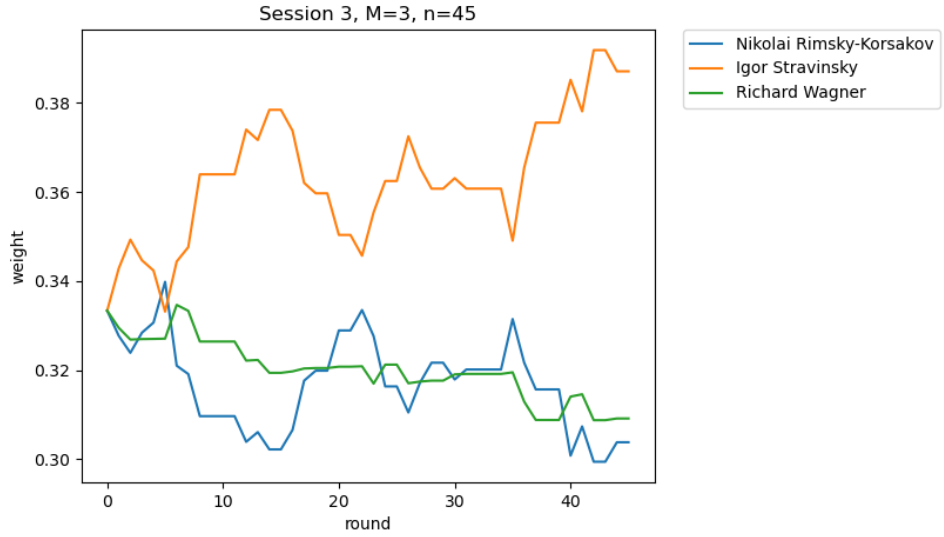

Figure 3: Evolution of the weights for the experts during my third session of Exp4 with the number of experts $M := 3$ and the number of rounds $n := 45$.

any of the actions with probability spectra that meaningfully differentiate their work. The small number of actions required means that the regret bound (proportional to the square root of the number of actions) can be kept small. There are many other possible sets of actions that could be used. The actions investigated in this work say nothing directly about tempo, harmony, texture, dynamic, expression, or about many other important aspects of music. I will investigate if other sets of actions can increase the usefulness of the Exp4 Algorithm while also keeping the regret bound small.

12 Conclusion

I find that the Exp4 Algorithm applied to algorithmic composition has reasonable regret bounds, seems to work well empirically for learning the preferences of a human user, and that it can produce useful results in a practical amount of time.

I find that by comparing my independent ranking of how well I like the composers used for experts with the rankings that Exp4 produced over three sessions totaling slightly less than an hour in duration, the hypothesis that the rankings produced by Exp4 are random has less than a 10% chance of producing the observed rankings.

In future work, I will investigate other sets of actions to see if they can increase the usefulness of the Exp4 Algorithm while also keeping the regret bound small.

Broader Impact

It is rare to be able to “experience" a learning algorithm in an intuitive setting, as one perhaps might in other fields, in an introductory Chemistry lab or on a Geology field trip. I hope that users will enjoy interacting with the Exp4 Algorithm through the web application that I developed, and that the web application will give users increased intuition for the performance of Exp4.

References

[1] Ffmpeg. URL https://ffmpeg.org/.

[2] Fluidsynth. URL http://www.fluidsynth.org/.

[3] mido. URL https://github.com/mido/mido.

[4] Amazon Web Services, Inc. Amazon web services (aws) - cloud computing services, 2020. URL https://aws.amazon.com/.

[5] Amazon Web Services, Inc. Aws lambda, 2020. URL https://aws.amazon.com/lambda/.

[6] Amazon Web Services, Inc. Amazon api gateway, 2020. URL https://aws.amazon.com/api-gateway/.

[7] Amazon Web Services, Inc. Amazon dynamodb, 2020. URL https://aws.amazon.com/dynamodb/.

[8] Amazon Web Services, Inc. Amazon s3, 2020. URL https://aws.amazon.com/s3/.

[9] Keunwoo Choi, György Fazekas, and Mark B. Sandler. Text-based LSTM networks for automatic music composition. *CoRR*, abs/1604.05358, 2016. URL http://arxiv.org/abs/1604.05358.

[10] Mary Farbood and Bernd Schöner. Analysis and synthesis of palestrina-style counterpoint using markov chains. In *ICMC*, 2001.

[11] Robert M Keller and David R Morrison. A grammatical approach to automatic improvisation. In *Proceedings, Fourth Sound and Music Conference, Lefkada, Greece, July."Most of the soloists at Birdland had to wait for Parker's next record in order to find out what to play next. What will they do now*, 2007.

[12] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. preprint, 2020. URL https://tor-lattimore.com/downloads/book/book.pdf.

[13] I-Ting Liu and Bhiksha Ramakrishnan. Bach in 2014: Music composition with recurrent neural network. *CoRR*, abs/1412.3191, 2014. URL http://arxiv.org/abs/1412.3191.

[14] Artemis Moroni, Jônatas Manzolli, Fernando Von Zuben, and Ricardo Gudwin. Vox populi: An interactive evolutionary system for algorithmic music composition. *Leonardo Music Journal*, 10:49–54, 2000. doi: 10.1162/096112100570602. URL https://doi.org/10.1162/096112100570602.

[15] La Scena Musicale. A few classical favorites in midi format, 1999. URL http://www.scena.org/midi/music/index.html.