Evolving Music from a Self-Organising Nanomagnetic Orchestra

Arthur Penty and Gunnar Tufte

Department of Computer Science,
Norwegian University of Science and Technology, Trondheim, Norway
arthur.penty@ntnu.no

Abstract

Artificial spin ice is a self-organising system of interacting nanomagnets which exhibits interesting and complex behaviour. In this paper we put the art in **art**ificial spin ice, presenting a novel mapping from a dynamical state trajectory to MIDI music for an ensemble of instruments. An evolutionary algorithm is used to search for new artificial spin ice geometries of higher musical quality, making use of Zipfian and entropic measures. Geometries of high fitness were discovered, and music resulting from the best geometry found is presented alongside this paper. Aside from the primary outcome of producing novel music, this unique viewpoint of artificial spin ice could allow for a more intuitive observation of its dynamical properties, interpreting their state trajectories through the medium of music.

Introduction

The rationale of extracting music from a dynamical system (DS) often takes one of two forms: to use music as a way to explore and gain insight into the behaviour and properties of a DS (Bilotta et al., 2005), or to use the DS as a seed to inspire or synthesise new music (Burraston and Edmonds, 2005).

Complex systems are systems in which the behaviour of the system emerges from the behaviour of the underlying interacting parts which make up the system. Complex systems are said to be self-organising when a large scale ordering arises from low-level interactions without any guiding centralised control. Complex systems seem a natural choice for making music as their behaviour can lie in a 'Goldilocks region' between chaos and simplicity. It is easy to make an analogy to music: music that is too simple is boring, and music that is purely random or chaotic is equally as unsatisfying to listen to. Koelsch et al. (2019) use predictive coding theory to describe how music listening is an active process where the listener is constantly making predictions about what they will hear next. As such, music must have some structure in order for meaningful predictions to be made, but not be so simplistic that it requires only trivial predictions.

Artificial Spin Ice (ASI), a class of nanomagnetic spin systems, have become a substrate of interest to both material physicists, studying its exotic metamaterial properties (Sklenar et al., 2019), and to computer scientists, exploring how its intrinsic self-organisational properties can be exploited for unconventional computing (Jensen et al., 2018). Jensen et al. (2018) show an ASI can exhibit a large variety of behaviours with regards to the number of terminal states produced by the ASI over a set of inputs, and that this behaviour can be tuned with relative ease. Given the richness and tunability of its behaviour, ASI seems a promising candidate for automatic music composition.

In this work we stimulate an ASI with an external field, and map the resulting state trajectory to music. We show how the geometry of an ASI can be evolved to improve the 'quality' of music produced using our mapping. Though the quality of a piece of music is subjective, we attempt to quantify this quality through methods that capture some of the phenomena which are present in aesthetically pleasing music.

Complex Systems and Music

Cellular automata (CA) are a simple example of a complex system exhibiting emergence and self-organisation. A number of studies have used CA to produce music through a variety of methods (Burraston and Edmonds, 2005). In general, a mapping is devised that transforms the time evolution of the CA into a musical representation (often MIDI). CA can be selected based on particular desirable properties such as a measure of complexity, or attempts can be made to optimise the CA for music production using optimisation techniques such as an Evolutionary Algorithm (EA).

Through sonificiation, the representation of data as sound, audio can provide an additional dimension to the observation of a system. Bilotta et al. (2005) translated certain quantitative features of a Chua's oscillator complex system into music. They found that human listeners where able to perceive many structures in the music which are present in the behaviour of the Chua's oscillator. They conclude that, through music, human cognitive abilities are able to analyse the complicated patterns produced by Chua's systems.

Putz and Svozil (2017) explore how elements and con-

cepts found in quantum mechanics can be expressed in music. Such an approach could provide a method to gain a more intuitive understating of some of the aspects of the notoriously unintuitive realm of quantum mechanics.

Artificial Spin Ice

ASI are complex systems consisting of many interacting nanomagnets, typically arranged on a 2D lattice. The nanomagnets are elongated along one axis, causing their spin to always align along this elongated axis. This allows us to consider the nanomagnet as a single binary spin. The state of the full ASI is therefore the ensemble of all the binary spins in the system. The spin of a magnet can change or 'flip' due to magnetic fields from its neighbouring magnets, or from stimuli such as an applied external field.

ASI exhibits self-organisation. Each magnet in the system can be viewed as an agent with the goal of minimising its energy. The energy of the magnetic agent depends on its spin and the spins of neighbouring magnets. As such there are competing interactions as the state which benefits one agent might be detrimental to another. These competing local interactions give rise to interesting large scale orderings.

The geometry of an ASI refers to the positions and orientations of the magnets in the system. The geometry plays a primary role in determining how magnets interact locally with each other and thus has a substantial effect on the large-scale behaviour of the ASI, emerging from these local interactions. This makes geometry an obvious parameter to vary in order to tune the ASI to have a desired property or behaviour.

Typically ASI require some stimulus to prevent the system from freezing (getting stuck in one state). Stimulus can be provided through perturbing the system with external magnetic fields, or through applying heat to the system. Temperature provides a stochastic stimulus to the system, causing magnets to flip randomly with a probability given by their energy state. An external magnetic field allows for a more controllable and targeted stimulus and as such is a good medium for encoding inputs to the system. The strength and angle of the external field can be adjusted.

Mapping ASI to Music

To create music from a given ASI geometry we must first obtain the time evolution of its state, which we do using the flatspin ASI simulator (Jensen et al., 2022). To prevent the dynamics of the system from freezing, we provide a stimulus in the form of an external global magnetic field. The field is applied with constant strength, but at each time step the angle at which it is applied is incremented by an angle θ. Through some preliminary experimentation, an angle $\theta = 23°$ was chosen, as it was found to have some effect on the system but not *too* much. Furthermore, as 23 is coprime with 360, the field angle can be stepped in this way many times before the angle repeats (359 times), the idea being

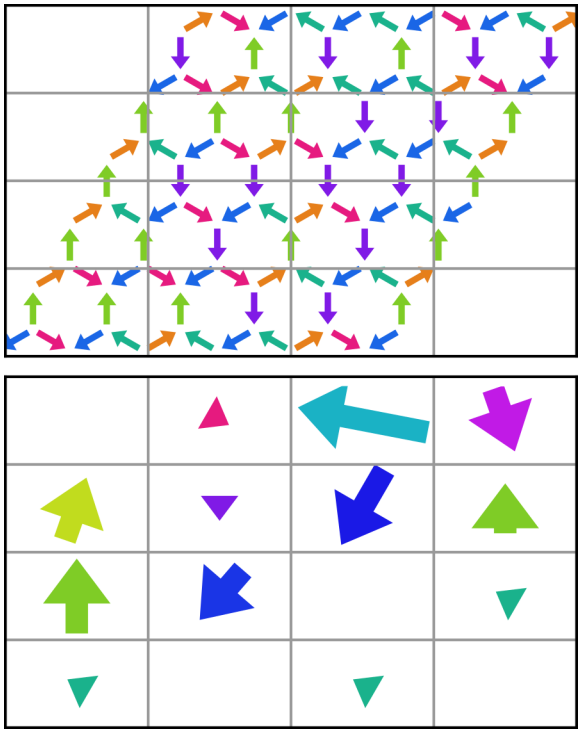

Figure 1: This figure shows how a grid of a given dimension (here 4x4) can be super imposed on the spin state of a ASI (top) and used to produce a locally aggregated view of the system (bottom) by summing the spins within each grid cell to obtain a macrospin. In some cases the cells in the aggregate view are empty because there are no magnets in those cells, whereas other cells are empty because the spins of the magnets within the cell cancel out.

that the constant changing of the stimulus is carried through to the music, preventing it from becoming too repetitive or monotonous.

From the flatspin simulation, we obtain a time series of the state of the system. In Fig. 1 we can see an example of a state. The figure also shows how we can superimpose a grid on to the state and sum the binary spins within each cell to obtain a aggregate spin or 'macrospin' for each cell. This is a common approach when visualising ASI as it can make large scale patterns more apparent (e.g. ferromagnetic or anti-ferromagnetic ordering).

Our mapping applies a grid on to the ASI and associates each cell with a different instrument. The state of a cell in a timestep t determines what note the corresponding instrument is playing, and as such the full system state represents the notes for an ensemble of instruments or orchestra.

Specifically we use three properties of a cell's spin to produce a note. We choose three properties of a MIDI note to control with the mapping: pitch, velocity and duration.

The pitch and duration of a note simply refer to how high or low a note is and for how long the note is sustained. The

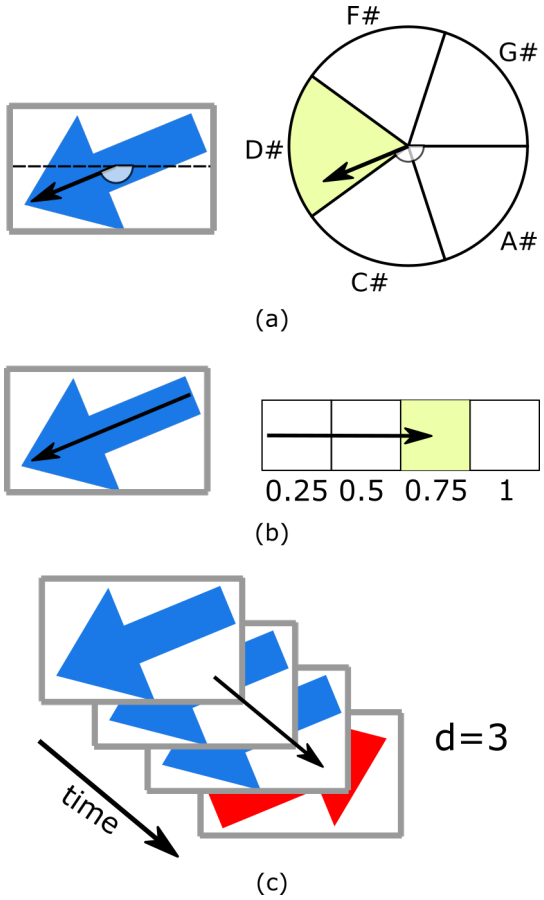

Figure 2: Different musical properties of a note are extracted from the a macrospin. (a) The pitch of the note is given by the angle of the macrospin which is discretized or rounded to a given scale. (b) The velocity of the note is given by the magnitude of the spin, and again is rounded into discrete bins. (c) The duration of a note is determined by looking along the time axis and observing how long the macrospin remains within the same pitch sector.

velocity of a note in MIDI terminology refers to how 'hard' a note is played, i.e., how hard a string is struck or reed is blown. Though velocity is most noticeable in the volume of a note, it can also be used to affect the timbre as it would on a physical instrument.

In our mapping, the angle of the macrospin determines the pitch of the note. The spin angle contains the most prominent information so its a natural choice to express through pitch, a salient feature in the resulting music. Furthermore, this gives a nice analogy between the correspondence and patterns in spins of the ASI, and the harmony (or disharmony) of the instruments in the produced music. The mapping must be supplied with a scale in the form of an ordered set of n_p pitches. We then partition a circle drawn around the origin of each macrospin into n_p discrete sectors. The pitch is then given by the sector in which the macrospin lies (Fig. 2a).

The velocity of a note is given by the magnitude of the macrospin. Again, this is quite a natural choice as it means when more of the sub-spins making up a macrospin align the note produced will be stronger, whereas, when the spins cancel each other, only a faint sound is produced. In this way the volume of an instrument expresses the energy of a system. As with the pitch, a list of n_v values defining the different velocities the system may produce is provided by the user. The interval given by the range of the macrospin magnitude is then evenly partitioned into n_v segments and used to determine the velocity (Fig. 2b).

The duration of a note is not determined by the macrospin in single timestep, but by looking across the time axis. To get the duration of a note we look forward in time too see how many timesteps must pass before the macrospin changes enough that the pitch will change. In effect this means consecutive notes of the same pitch on the same instrument are grouped into one long held note. The results of this is that systems or macrospins with fast changing dynamics will produce many short quick notes, while slower dynamics will produce long droning notes. We set an upper bound on how long a note is held $d_{\max}$, after which the instrument remains silent until a note of a different pitch is played (Fig. 2c). A notable limitation that follows from this is that an instrument cannot play two separate notes of the same pitch consecutively.

Evolution

EAs are a class of bio-inspired optimisation methods which use the Darwinian concepts of variation and natural selection to provide solutions to certain problems. EAs are particularly useful when there is no obvious gradient to follow in the performance of a task, e.g., *"how should I tweak this magnet to make my song sound better?"*. The three necessary components to make use of EA are: a population of individuals representing a solution to the given task, a method to mutate or make small changes to an individual, and a method to compare or evaluate the fitness of an individual at the given task.

The individuals in our population are ASI geometries. We have previously shown EAs to be a useful tool in finding new ASI geometries with desired properties (Penty and Tufte, 2021, 2022). In contrast to the tile-based approach used previously, we use a simpler, lattice based approach to represent the full ASI geometry. We now describe the process of mapping an individual to its geometry. This process is illustrated in Fig. 3.

The positions of the magnets in the ASI geometry are the points in the 2D lattice spanned by two basis vectors b_0 and b_1. To allow for a greater diversity of geometries we tile a binary matrix H over the lattice to determine which magnets are 'expressed'. Finally a matrix of angles A is tiled

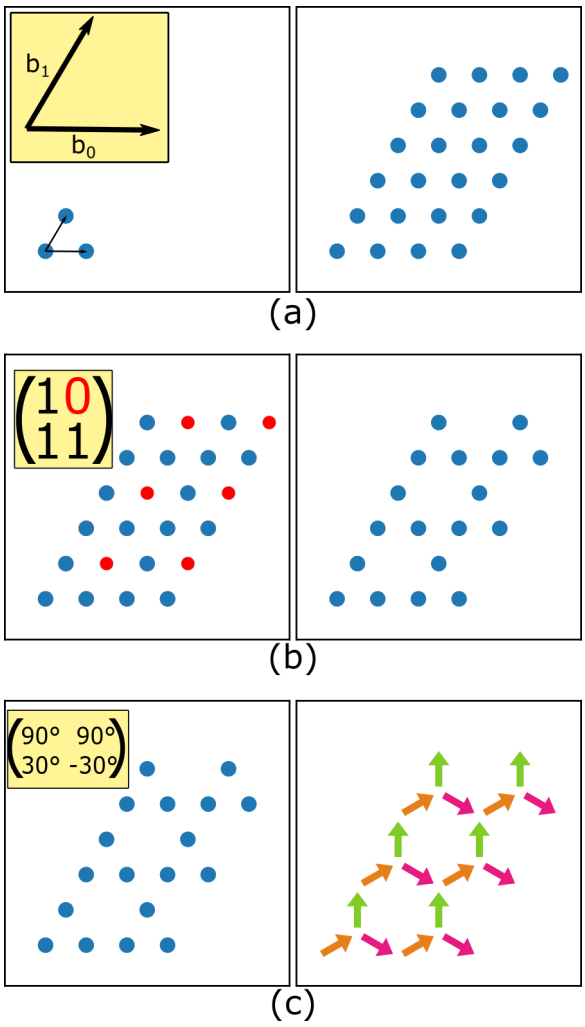

Figure 3: The process described in section "Evolution" of building a geometry from the basis vectors b_0, b_1, hole matrix H and angle matrix A. The first step (a) is to repeatedly apply the basis vectors starting from the origin until be have a lattice of the desired size. (b) shows how the hole matrix is tiled over the lattice. Any lattice points that align with a zero in the hole matrix are removed. (c) The angle tile is then tiled over the lattice to determine the angle of the magnet to be placed at each lattice point.

over the lattice to determine the angle of the magnet at that position. Each individual in the population consists of two basis vectors b_0 and b_1, a hole matrix H and an angle matrix A.

Mutation of an individual is achieved through randomly selecting one of its components to modify. In the case of the basis vectors and angle matrix, a Gaussian mutation can be selected where Gaussian noise with a mean of zero is added to the selected vector or matrix. Both the angle matrix and hole matrix can be modified using a swap mutation where two elements in the same matrix have their values swapped. The hole matrix can be mutated with a bit flip point mutation where an element has its binary value flipped. Finally, both matrices can be mutated to have their size in one of their dimensions altered. For further variation, we implement a simple crossover operator that can create new individuals as offspring by mixing two parent individuals. Offspring sample their attributes (vectors and matrices) randomly from their two parents.

As stated in section "Mapping ASI to Music", the ASI is driven by an external magnetic field of constant strength, with a varying angle. A suitable value for this field strength is highly dependent on the geometry of the ASI it is being applied to. Due to this, we also place the strength of this field under evolutionary control. Each individual possesses its own field strength parameter which can be selected for Gaussian mutation. In crossover offspring randomly inherit the field strength value from one of their parents.

While creating and modifying individuals we place constraints on their parameters. The shape of the angle matrix is $m_A \times n_A$ such that $0 < m_A, n_A \leq 5$, and the shape of the hole matrix is $m_H \times n_H$ such that $0 < m_H, n_H \leq 7$. The lengths of the basis vectors is in the range $[0.5, 1.1]$ to ensure the magnets are not too far away (resulting in too weak interactions) or too close together (overlapping magnets). We also ensure the angle between the two basis vectors remains in the interval $[45°, 180° - 45°]$ so that the lattice does not become too sheared and to prevent the bases from becoming parallel. We constrain the hole matrix to have no more than 70% of its elements as 0s to prevent empty or very sparse lattices. Finally, we constrain the field strength of the external stimulus field to be in the interval $[0.03, 0.27]$.

The flatspin parameters used for the simulation were disorder = 0.1 and alpha = 0.01. All other parameters used the default values as of flatspin version 2.2. For an explanation of the flatspin parameters and their default values, please refer to the flatspin documentation (flatspin contributors, 2022).

Musical Fitness Function

Creating a formula to fully describe the quality of a given piece of music is extremely difficult and likely impossible due to the inherent subjectivity of music quality. Such is the difficulty in creating a fitness function, some attempts

at evolving music rely on humans for evaluation (Biles et al., 1994). But human-in-the-loop evolution, while neatly sidestepping the problem of implementing a quantitative measure for musicality, suffers from slowness and inconsistency. The alternative is to use heuristics as a surrogate for a true musical fitness, to guide evolution towards regions where musicality is more likely to be found.

The fitness function we devise to assess the quality of music expressed in an individual's state trajectory is made of two heuristics. For the first heuristic we measure how Zipfian the produced music is. Zipf's law is a statistical relationship first observed in the word frequency of natural languages (Zipf, 1949). It states the frequency with which a certain occurrence is observed is inversely proportional to the rank of that occurrence in order of frequency. Specifically, in terms of language, it means the most common word in a text occurs twice as frequently as the second most common word and thrice as frequently as the third most common word, and so on. Zipf's law has been observed in many fields, including the arts. Manaris et al. (2003) compiled distributions from various aspects of a large corpus of music and found Zipf's law to be a necessary condition on certain dimensions of the music, though they stress it alone is not sufficient for aesthetically pleasing music. This has lead to Zipf's law being used or suggested as fitness function for evolutionary music composition in a number of works (Manaris et al., 2005; NCRA, 2015; Kirke and Miranda, 2007; Jensen, 2011).

Specifically in our case we calculate the Zipfian error of three aspects of our music, the distribution of pitches, the distribution of note velocities and the distribution of note length. In each of these cases we calculate the Zipfian error by ordering the frequencies and normalising such that the largest frequency becomes 1, then we sum the absolute difference between each frequency and the frequency that would be expected given the idealised Zipfian distribution. This is formulated as:

$$E_z(s_1, \ldots, s_N) = \sum_{i=0}^{N} |\frac{s_i}{s_1} - \frac{1}{i}|$$

where $(s_1, \ldots, s_N)$ is a sequence of frequency counts in decreasing order.

As mentioned in Manaris et al. (2003), just following Zipf's law is not the only requirement for satisfying music. From our preliminary studies of using only the Zipfian loss metric as a fitness function, one stark deficiency noticed was a tendency to produce music where all instruments were playing the same notes at the same time, which led to the produced music sounding quite boring. This deficiency is not captured in the Zipfian metrics used as if one instrument is playing a stream of Zipfian notes, then this stream can be replicated on all instruments without incurring any loss from the Zipfian metric.

To remedy this, we introduce a second heuristic, the joint entropy of the notes played over each instrument. Joint entropy gives a measure of the uncertainty of different parts of a system in relation to each other. In our case we can think of the joint entropy as measuring a form of correspondence between the instruments, e.g., if knowing what note the violin is playing lets you predict with high confidence what notes the cello and viola are playing, then these would be said to have low joint entropy as only a small amount of information allows you to infer the full state of all variables. The joint entropy is given by

$$H(X_1, ..., X_n) = \\ - \sum_{x_1 \in \mathcal{X}_1} ... \sum_{x_n \in \mathcal{X}_n} P(x_1, ..., x_n) \log_2[P(x_1, ..., x_n)] \quad (1)$$

We can estimated $P(x_1, ..., x_n)$ as the number observations of $(x_1, ..., x_n)$ divided by the total number of all observations.

Given these two heuristics we can construct a fitness function. We create normalised versions of the two heuristics $\bar{E}_z$ and $\bar{H}$ such that they produce values in the range $[0, 1]$. For the Zipfian error this is done by dividing by the maximum possible error, which is attained when all the values of the sequence are equal. The normalised entropy measure $\bar{H}$ is obtained by setting the base of the log in Eq.1 to the largest possible number of unique occurrences $(x_1, ..., x_n)$. Our fitness function f can then be defined as

$$f(P, V, D, X) = \bar{H}(X) - \frac{1}{3}[\bar{E}_z(P) + \bar{E}_z(V) + \bar{E}_z(D)]$$

where P, V and D are the rank distributions for the produced music's pitches, velocities and durations respectively aggregated over all instruments. X is the series $X_1, \ldots, X_N$ where each X_i is the scale from which the i^{th} instrument's pitches are taken. By multiplying the sum of the Zipf metrics by one third, we obtain a fitness function that equally weights the importance of the Zipf metric and the entropy metric. The result of applying the fitness function is a a value in the range $[-1, 1]$ where 1 is the perfect score and -1 is the worst possible score.

We set up our EA using the individual representation, mutation operators and crossover operator as defined in section "Evolution", and with the fitness function defined in section "Musical Fitness Function". A population is created by generating 100 randomly initialised individuals. For each generation of the EA new individuals are created using the mutation operators and crossover operator. We use a relatively large mutation rate and crossover rate of 50%, though in both cases we retain a copy the unmodified parent individual in the population. At the end of each generation, selection occurs. The fitness of all individuals is measured, and individuals not in the top 100 are discarded.

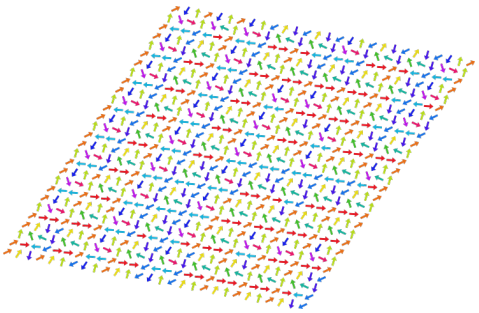

Figure 4: The ASI geometry of highest fitness in the final generation of the evolutionary run.

The EA was run for 300 generations, with individuals grown until they contained at least 200 magnets. For the music mapping a 12-note C major scale was supplied, the list $[0.2, 0.4, 0.6, 0.8, 1]$ was used as the possible values the velocity could take, and the maximum note duration $d_{\max}$ was set to 8. A 3×3 grid was used to create the macrospins, resulting in an ensemble of 9 instruments.

Results

The results of this evolutionary run are shown in Fig. 4 & Fig. 5. From the top-left panel of Fig. 5 we see that the best fitness in the generation started out quite high at 0.56 and quickly increase up to 0.895 at generation 22 at which it peaks and is then constant for the remaining generations. Often in EAs a fast increase in fitness followed by long periods of no improvement is a symptom of premature convergence, where the EA over-fits and the population loses diversity. Though the other panels in Fig. 5 appear to indicate there is still diversity in the population, at least phenotypic diversity.

We can also see the Zipfian error metric is consistently low and as such the change in fitness seem to be mostly driven by the change in the entropy metric. This can be seen very clearly in the two panels on the right in Fig. 5, which show the fitness very closely mirrors the joint entropy metric. The mean entropy metric and thus the mean fitness can be seen to fluctuating greatly. This likely indicates that the mutation on crossover operators have a high probability of deleterious effect or the effect they have on the phenotype is too large.

The best individual in the final generation was used to create MIDI music, the instruments used were randomly sampled from a subset of the set of general MIDI instruments, with some of the more unusual 'instruments' excluded (dog barks and telephone rings etc.). Each instrument was also randomly assigned the octave for the first note of its scale. We also use the Microsoft GS Wavetable Synth to synthesise the MIDI file into an audio file, to gives a device independent way to listen to the music as, unlike the MIDI file, it sounds the same on all devices. This music can be listened to at: `https://osf.io/h7tcd/`.

Also included in this repository is a manually 'mastered' version of the music to give an idea of how the music could sound with some minimal input from a human (setting the master volume level for each instrument, choosing the octaves and using a more modern virtual instruments of the their given type). A benefit of producing music in this way is that it also provides a natural music video, or visualisation. For both the raw music version and the manually mastered version we include a video showing the state of the system that is producing the music alongside the aggregated state.

Listening to the music, we can with some confidence claim that we achieved our baseline goal of creating music that is somewhere between too simple and too chaotic. The music certainly has some kind of structure to it, we hear patterns of repeated pitches and rhythms. Due to the changing angle of the field and the internal memory of the system, the music is never repeating itself exactly. Though, as the music goes on there is some sense that the music is 'treading water' becoming slightly tedious. This could be due to a flatlining of the musical tension in the piece.

Tension is an abstract, hard to formulate, concept from music theory which describes the perceived increase and release of unrest or anticipation in a piece of music (Farbood, 2012). The tension is a complex function of many musical dimensions including pitch, volume, rhythm and speed. Not considering the tension during composition can lead to monotonous or tedious sounding music. In our system the lack of musical tension could arise from the energy of the system being too similar over time, or a lack of richness in the input stimulus.

Conclusion

In this work we have laid out a method of extracting music from ASI and a means of evolving them in search of aesthetically pleasing music. The mapping was able to produce interesting musical patterns and a variety of different behaviour over different instruments.

In terms of the fitness function chosen, the results were of high quality. Though the quick speed the EA reached this fitness indicates that the fitness used maybe too lenient. Furthermore, while there is not much room for improvement in the achieved fitness, the musical output of the systems could certainly be improved upon. These observations point to the need to further develop the fitness function to take into account other musical aspects such as the tension. Alternatively, different stimuli could be explored. By swapping between different stimulus style, a chorus-verse like structure could be imprinted, which may help to break up the monotony the system appears to tend towards.

While the mapping produced the notes to play automatically, there were still aspects such as instrument choice and octave range that had to be chosen either at random or by a human. An obvious extension to this work would be to

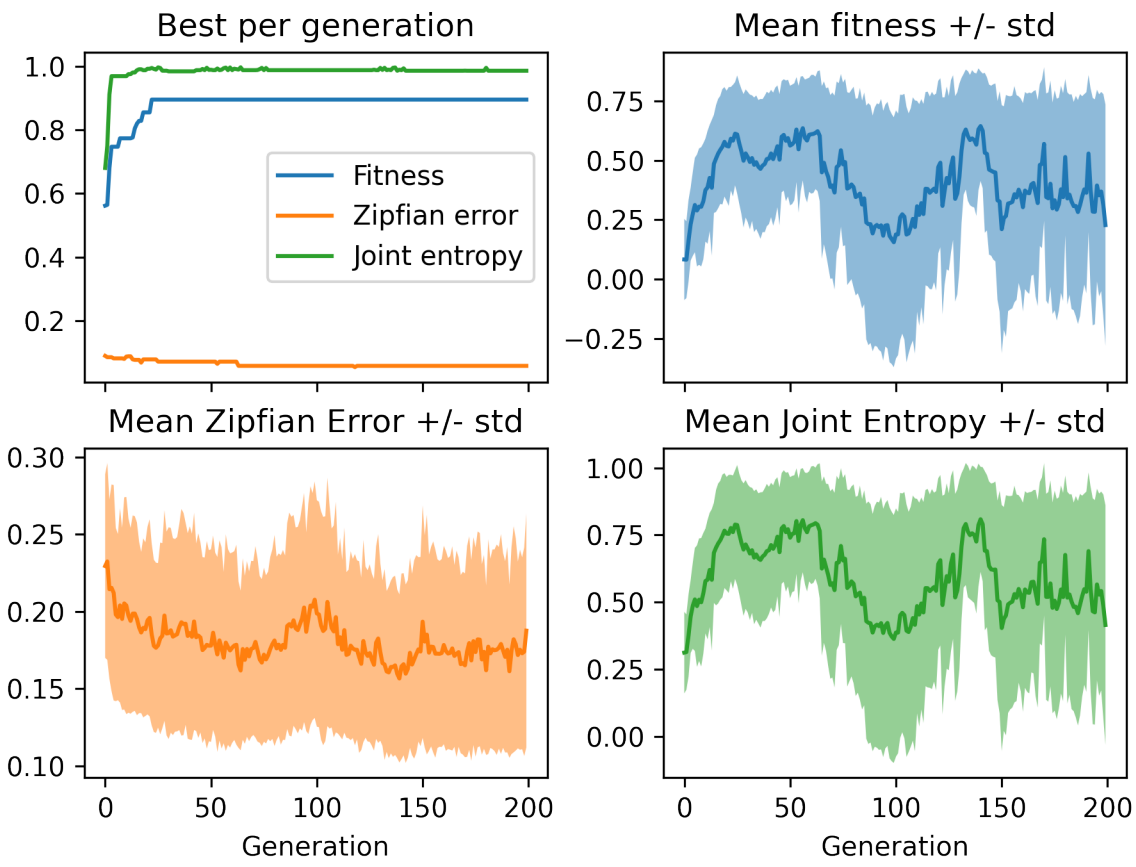

Figure 5: In the top-left plot we see for each generation the maximum fitness, maximum joint entropy measure, and minimum Zipfian error. The remaining graphs show how the mean and standard deviation of the three properties changed over the course of the run. Though the algorithm was run for 300 time steps we show only the fist 200 as after the 150^{th} generation the algorithms behaviour does not change substantially.

somehow put these choices under the control of the system. Additionally, more features could be extracted from the ASI simulations to control other aspects of the produced MIDI such as pitch bend or to add percussion.

One necessary design choice was choosing how to drive the system. Even if constrained to interacting with the system using a global field, there are many different input schemes that could be used, each giving a different musical interpretation of an ASI geometry. As such, evaluation of an ASI geometries musical quality, or investigating its properties via music, is always done with respect to the chosen input encoding. To investigate the ASI geometry more generally, several input encoding may need to be evaluated. Alternatively, the dependence on the driving field can viewed as an avenue for user interaction. Real time manipulation of the driving field could allow an ASI geometry to be "played" as an instrument.

Another direction for future study is to use such a mapping to perceive the physical properties of an ASI via observation of its state trajectory. We already mentioned that musical tension could be an analogy for, amongst other things, the energy present in an ASI. But, in a much broader scope, the richness of the state trajectory of an ASI, or DS in general, is of interest for many different reasons. Through an appropriate mapping of the state trajectory to music, it may transform the problem evaluating a state trajectory to a more human-friendly problem of evaluating the quality of music. This could allow for humans to get an intuitive sense of the richness of DS's state trajectory or even where its deficiencies may lie.

This work presents the first step into the domain of nanomagnetic music, a fusion of cutting edge material science, physics and music.

Acknowledgements

This work was funded in part by the Norwegian Research Council IKTPLUSS project SOCRATES (Grant no. 270961), and in part by the EU FET-Open RIA project SpinENGINE (Grant no. 861618). flatspin (Jensen et al., 2022) simulations were executed on the NTNU EPIC compute cluster (Själander et al., 2019).

References

Biles, J. et al. (1994). Genjam: A genetic algorithm for generating jazz solos. In *ICMC*, volume 94, pages 131–137. Ann Arbor, MI.

Bilotta, E., Gervasi, S., and Pantano, P. (2005). Reading complexity in chua's oscillator through music. part i: A new way of understanding chaos. *International Journal of Bifurcation and Chaos*, 15(02):253–382.

Burraston, D. and Edmonds, E. (2005). Cellular automata in generative electronic music and sonic art: a historical and technical review. *Digital Creativity*, 16(3):165–185.

Farbood, M. M. (2012). A parametric, temporal model of musical tension. *Music Perception*, 29(4):387–428.

flatspin contributors (2022). flatspin documentation. Available at `https://flatspin.gitlab.io/`.

Jensen, J. H. (2011). Evolutionary music composition: A quantitative approach. Master's thesis, Institutt for datateknikk og informasjonsvitenskap.

Jensen, J. H., Folven, E., and Tufte, G. (2018). Computation in artificial spin ice. In *Artificial Life Conference Proceedings*, pages 15–22. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info ....

Jensen, J. H., Strømberg, A., Lykkebø, O. R., Penty, A., Leliaert, J., Själander, M., Folven, E., and Tufte, G. (2022). flatspin: A large-scale artificial spin ice simulator. *Physical Review B*, 106(6):064408.

Kirke, A. and Miranda, E. (2007). Evaluating mappings for cellular automata music. In *Proceedings of ECAL Workshop on Music and Artificial Life*. Citeseer.

Koelsch, S., Vuust, P., and Friston, K. (2019). Predictive processes and the peculiar case of music. *Trends in Cognitive Sciences*, 23(1):63–77.

Manaris, B., Machado, P., McCauley, C., Romero, J., and Krehbiel, D. (2005). Developing fitness functions for pleasant music: Zipf's law and interactive evolution systems. In *Applications of Evolutionary Computing: EvoWorkkshops 2005: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, and EvoSTOC Lausanne, Switzerland, March 30-April 1, 2005 Proceedings*, pages 498–507. Springer.

Manaris, B., Vaughan, D., Wagner, C., Romero, J., and Davis, R. B. (2003). Evolutionary music and the zipf-mandelbrot law: Developing fitness functions for pleasant music. In *Applications of Evolutionary Computing: EvoWorkshops 2003: EvoBIO, EvoCOP, EvoIASP, EvoMUSART, EvoROB, and EvoSTIM Essex, UK, April 14–16, 2003 Proceedings*, pages 522–534. Springer.

NCRA, U. C. (2015). Grammatical evolution with zipf's law based fitness for melodic composition.

Penty, A. and Tufte, G. (2021). A Representation of Artificial Spin Ice for Evolutionary Search. *ALIFE 2021: The 2021 Conference on Artificial Life*. 99.

Penty, A. and Tufte, G. (2022). Evolving artificial spin ice geometries towards computing specific functions. *Evo* 2022*, page 22.

Putz, V. and Svozil, K. (2017). Quantum music. *Soft Computing*, 21:1467–1471.

Själander, M., Jahre, M., Tufte, G., and Reissmann, N. (2019). EPIC: An Energy-Efficient, High-Performance GPGPU Computing Research Infrastructure. *arXiv:1912.05848 [cs]*.

Sklenar, J., Lao, Y., Albrecht, A., Watts, J. D., Nisoli, C., Chern, G.-W., and Schiffer, P. (2019). Field-induced phase coexistence in an artificial spin ice. *Nature Physics*, 15(2):191–195.

Zipf, G. K. (1949). Human behavior and the principle of least effort: An introduction to human eoclogy.