Rhythmic Density for Sonifying Quantum Superpositions

Walker Smith[1] , Dmitri Volkov[1] , Alex Alani[1]

[1]Center for Electronic and Computer Music – Indiana University Bloomington
107 S. Indiana Ave. - 47405 - Bloomington - Indiana - United States of America

walksmit@iu.edu, dvolkov@iu.edu, alealani@iu.edu

Abstract. In this paper we introduce a novel approach for sonifying a superposition of quantum states, by using rhythmic density (e.g. faster rhythms) to represent probability amplitudes. In doing so, we hope to help listeners obtain an intuitive understanding of the probabilities behind quantum systems. We start with a brief overview of our approach, then demonstrate how it can be used with several implementations. We explain the advantages of rhythmic density over other approaches, and compare advantages and disadvantages of our work with other quantum system sonification schemes. The paper concludes by outlining areas for future development and research.

1 Introduction

Quantum computation is an emerging field with many promising applications across a wide variety of disciplines[1]. However, it is often tricky to gain an intuitive understanding of how quantum computation (and quantum systems in general) work in practice[2]. That said, music has a rich history of explaining difficult concepts in intuitive ways [3][4], dating back at least as far as Johannes Kepler[5]. By aligning unintuitive concepts of quantum computation with intuitive concepts in music, we hope to enable a wider audience to understand the underpinnings of quantum computation, as well as give experienced quantum practitioners a new tool for analysis. In this paper, we introduce a new paradigm for sonification of quantum systems through analogizing higher amplitudes of a state with greater rhythmic density (e.g. faster rhythms).

1.1 Quantum Superposition

Arguably, the primary difference between classical information and quantum information is that quantum information can exist in superposition. As such, we choose to focus on sonifying superpositions of quantum states. By focusing on superpositions, we provide a solid bedrock for future work (briefly explored at the end of the paper) for sonifying other aspects of a quantum system, including entanglement and phase.

1.2 Rhythmic Density and Probability

Our approach to sonifying a quantum state is as follows: first, calculate the probability of measurement for each of the basis states, then use these probabilities as input to one of several rhythmic density based sonification algorithms. These sonification algorithms map a higher probability to higher rhythmic density with faster rhythms and more notes, while lower probabilities correlate to lower rhythmic density with slower rhythms and less notes. As we outline later in the paper, rhythmic density is highly quantifiable for the human ear compared to other potential "sonification dimensions," and has other advantages as well. While our approach is infeasible to run on quantum hardware (because a useful approximation of the probabilities of a quantum state on hardware requires many invocations of the same program), it is still very useful with smaller simulated quantum systems as a pedagogical tool, analytical too, and creative tool for musicians.

2 Implementations

To sonify a quantum circuit, we first need to extract parameters from it which we then can translate into sound. Due to the nature of quantum physics, it is not possible to extract information from a quantum system without disturbing it; furthermore, because a quantum system collapses upon measurement, some information is lost. For this reason, we chose to use a simulated quantum system through (Qiskit[6] and MicroQiskit[7] though Max[8]).While it technically is possible to adapt our method to run on quantum hardware, it would be computationally expensive, especially if one wants to hear how the state changes as it progresses through state transformations. As mentioned, it would require a large number of shots (e.g. quantum program invocations) to find a good approximation of the quantum state at a single stage[9]. If one wanted to sonify the state as it progresses through state transformations, it would be necessary to perform the same number of shots at each stage in the computation, which would multiply the number of necessary shots. Furthermore, if the program being sonified itself requires a measurement operator, post-selection would be necessary.

By using a simulated quantum computer, it becomes possible to accurately extract the statevector at each step in a quantum computation without disturbing the system[10]. This gives us the probability amplitude for each quantum basis state. From this, we can calculate the probability of the measurement of each basis state by multiplying each probability amplitude by its complex conjugate. This gives us a single real number for each state, which we can feed into our sonification algorithms. While we do lose information about the phase, we can consider this as a separate parameter for each state. The sonification algorithms then generate note data, which then can be processed

into sound data which is the final sonification.

2.1 Python Implementation

One implementation of this technique uses Python[11] and the Qiskit[6] framework for working with quantum computers. We define a function, get_probabilities_at_steps, which takes a user-defined Qiskit QuantumCircuit object and returns a list which has the statevector of the quantum system after each gate in the quantum circuit defined by the QuantumCircuit object. The implementation of this function simply iterates through the instructions from the user-provided QuantumCircuit and applies them to a fresh QuantumCircuit, using a Qiskit builtin to get the statevector of the fresh QuantumCircuit at each step. Another Qiskit builtin is then used to calculate the measurement probabilities for each basis state for each statevector.

The sonification approach used by our Python implementation transforms each probability into a rhythm by figuring out the ratio of each probability to the lowest probability, and giving each state a rhythm based on this ratio. This results in a polyrhythm between the notes, with ratios corresponding to the probabilities.

The Python implementation is freely available via Google Colab[12]. To use it, simply run the code blocks in order. This installs the necessary dependencies, defines helper functions, generates the circuit, and outputs a MIDI[13] file for the sonified circuit! The MIDI data can then be processed into audio data by a digital audio workstation, such as Logic Pro[14] or Bitwig Studio[15]. The circuit can be tweaked freely, using standard Qiskit conventions, in the 3rd code block. To download the MIDI file updated for the tweaked circuit, simply re-run the 4th code block.

The implementation can be accessed at the following link: https://colab.research.google.com/drive/1Es9EgHPlMFIzgL5aJfddSCUdq9vngosS?usp=sharing .

2.2 Max Implementation

The Max implementation is built atop the QAC Toolkit[16] package, which in turn is built atop MicroQiskit[7]. A circuit is defined in real-time by sending messages to an och.microqiskit object. Each message corresponds to the application of a quantum gate. We then use another message to extract the statevector. We send the statevector, along with the index of the desired state, to subpatchers which extract the corresponding probability amplitude for each state and convert them into probabilities. The outputs of each of these subpatchers can then be used as the input for a sonification method. Note that the circuit and subpatchers are constructed such that when any new gate is applied, the information propagates immediately through the system, without requiring any additional messages.

To use the Max implementation, download and open the Max file. Define messages which feed into the och.microqiskit object to perform the desired gates and operations on the quantum circuit. When ready to run, lock the circuit and press the box corresponding to each message to apply its operation to the quantum circuit. The sonification should update accordingly in real-time with each message sent!

3 Why Rhythmic Density

As mentioned, we chose rhythmic density as our "sonification dimension" of choice for superposition. Rhythmic density here refers to how many notes occur within a uniform span of time. For example, a span of a second with 5 notes has greater rhythmic density than a span of a second with a single note. Note that this definition holds even if the note durations within the span are not the same. To differentiate between different basis states, each state is assigned a different pitch or timbre.

To achieve a more quantitative representation of the probabilities of states, we correlate the probabilities of each quantum state with higher rhythmic density for the pitch/timbre assigned to each state. According to our paradigm, a pitch corresponding to a state with a higher probability of being measured will produce more attacks in a unit of time than a pitch corresponding to a state with a lower probability. In other words, measurement probability is positively correlated with rhythmic density.

For example, consider a single qubit in an equal superposition state, $|+\rangle = \sqrt{\frac{1}{2}}|0\rangle + \sqrt{\frac{1}{2}}|1\rangle$. Here, the measurement probabilities of $|0\rangle$ and $|1\rangle$ are both $\frac{1}{2}$, so they are the same. Because both measurement probabilities are the same, both states should have the same rhythmic density. With straightforward sonification methods, this would mean that the rhythms corresponding to both states would be the same (Figure ??)[1].

On the other hand, consider the state $\sqrt{\frac{1}{4}}|0\rangle + \sqrt{\frac{3}{4}}|1\rangle$. Here, the probability of measurement for the $|0\rangle$ state is $\frac{1}{4}$, and the probability of measurement for the $|1\rangle$ state is $\frac{3}{4}$. So the $|1\rangle$ state is three times as likely as the $|0\rangle$ state to be measured, and thus should have three times the rhythmic density. As such, with direct sonification methods, the ratio of the rhythms corresponding to the $|0\rangle$ and $|1\rangle$ states would be $1:3$ (Figure ??).

[1] At the time of writing, we were not able to produce figures and audio examples due to extenuating circumstances. We plan updating the paper to include them as soon as possible

3.1 Rhythmic Density vs. Loudness

Existing approaches to sonifying superposition have used loudness to represent a higher measurement probability[17]. While this is attractive due to enabling a direct mapping between probability amplitudes and sound wave amplitudes, in reality it is difficult for the human ear to recognize precise differences in loudness[18].

To hear an audio representation of this principle and how it compares to correlating probabilities to amplitudes, first consider audio excerpts 1-3, which each represent a qubit in a unique superposition of: $\sqrt{x}|0\rangle + \sqrt{y}|1\rangle$, with two sine tones with frequencies corresponding to C5 and E5 for $|0\rangle$ and $|1\rangle$, respectively. The ratio of loudness C5/E5 is determined by the ratio ratio x/y. The reader is encouraged to listen to each audio excerpt and, from the loudness of each tone, predict the approximate values of x and y. In other words, try to identify the louder tone (if any), and how much louder one tone is than the other.

We expect the reader to find this to be a challenging task. Even if one could identify the louder tone, it is very difficult to tell how much louder it is. A rhythm-based approach should be easier. Audio excerpts 4-6 represent the same three superpositions with the same three probability amplitudes, but this time the ratio of x/y is sonified as rhythmic density. The tones are not constant but rather repeat at specific time intervals, and the rate of repetition is determined by the probability ratio x/y. Now, try to approximate the ratio x/y for each superposition. In other words, try to identify which note is being repeated faster (if any), and how many times faster that note is being repeated. This exercise will be familiar to music students forced to do rhythmic dictation in ear-training classes.

Even without the pleasure of taking an ear-training course, it is likely easier to identify the relative probabilities in excerpts 4-6 than in 1-3. The easily quantifiable nature of rhythmic values and note-length-relations compares favorably to the extreme difficulty of comparing loudness. Notating audio excerpts 4-6 in standard musical staff notation makes this clearer (Figure ??).

Consider a more complicated example: a two-qubit system in a superposition of $\sqrt{x_1}|00\rangle + \sqrt{x_2}|01\rangle + \sqrt{x_3}|10\rangle + \sqrt{x_4}|11\rangle$, with $x_1 : x_2 : x_3 : x_4$ equal to $1 : 2 : 3 : 4$. In other words, a superposition defined by $\sqrt{\frac{1}{8}}|00\rangle + \sqrt{\frac{1}{4}}|01\rangle + \sqrt{\frac{3}{8}}|10\rangle, \sqrt{\frac{1}{2}}|11\rangle$. We can arbitrarily assign the pitches C5, E5, G5, and Bb5 to the states $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$, respectively. Audio excerpt 7 provides a realization of this rhythmic result, and figure ?? provides a score notation.

Discerning the rhythmic relationships is certainly more challenging here, but for a loudness-based approach it would be virtually impossible. Audio excerpt 8 provides such a realization, and it is immediately clear that it is difficult to even rank the loudness of all 4 pitches, let alone determine the precise loudness relationships.

3.2 Incorporating Pitch and Timbre

For our approach to be useful in more complicated systems, we not only adopt this probability-rhythmic density idea, but also represent each state with a unique instrument. Thus, the combination of instrument and pitch is unique for every state. In this case, instrument refers abstractly to the pitch being played with a unique timbre. Audio excerpt 9 is the same as audio excerpt 7, except each state has been assigned to a different instrument.

Note that it is important for the instruments corresponding to each quantum state to be easily differentiable for listeners. This can be accomplished by assigning each instrument to a distinct region in musical pitch space (e.g. for a 4-state system, the four instruments might be in bass, tenor, alto, and soprano regions). If the states are in different, non-overlapping frequency ranges, then frequency masking (e.g. when one sound makes it hard to hear another because the frequencies overlap) is unlikely to occur, making it easy to tell the instruments apart[19]. Timbre can also play a large role: different timbres can be accomplished by using different synthesizer waveforms, or different instruments, ideally from different families (e.g. for a 4-state system: bass clarinet, viola, trumpet, and xylophone, which is an ensemble we feel is underrepresented in much of the classical repertoire).

3.3 Incorporating Motives

By assigning each instrument a pitch space instead of a single pitch, we unlock the option to use musical motives or lines instead of just rhythms. The motives (or motifs, as the two terms are often used interchangeably) are separated in pitch space, and have their own melodic contour. Motives are flexible, and can be used to represent individual states, or measurement probabilities of a state. For example, a different motive can be assigned to each state, with the speed at which each motive is performed corresponding to the measurement probability. Alternatively, a set of motives with low to high rhythmic density can be used to represent the measurement probability of a basis state, with each state differentiated by pitch space and timbre. Motives are often more interesting to listen to than repeated notes (citation needed), and as such greatly expand the utility of our approach as a creative tool.

4 Prior Work

4.1 qMuVi

Several effective sonification models have been applied to representation of quantum computers. The

qMuVi[17] project, developed by researchers at the University of Melbourne, sonifies quantum states in N-qubit systems by assigning each of the 2N states to a unique pitch. For example, a 2-qubit system with the 4 measurement basis states $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ can be represented by 4 distinct, arbitrary pitches. A superposition of multiple states thus sounds like a chord with multiple notes, where the loudness of each note is related to its probability.

This is an intuitive musical mapping, but it suffers from a problem of quantitative representation of the respective probabilities as outlined above. Additionally, discerning loudness differences in superimposed tones can be difficult, especially if those tones are far apart in frequency-space, due to the uneven perception of loudness across the frequency spectrum[19]. We have found that our rhythmic density based approach can more accurately represent the quantitative relationships between probability values in a superposition state, by correlating probability to rhythmic density instead of loudness.

4.2 Q1Synth

Another approach to sonifying a qubit is demonstrated by the Q1Synth[20], developed by the ICCMR at University of Plymouth. Besides sonification, the Q1Synth also provides visualization of a qubit in the form of a Bloch sphere, which shows the amplitudes and phases as a position on a three-dimensional sphere. The point on the top of the sphere represents the $|0\rangle$ state, and the point on the bottom represents the $|1\rangle$ state; points in the middle represent linear combinations of the two states, with a phase indicated by where along the sphere they lie. The general sonification approach of the Q1Synth is to take the polar coordinates of the position of the point of the Bloch sphere and use these as parameters for one of several synthesis methods, including FM synthesis, additive synthesis, and granular synthesis.

Q1Synth presents two ways for users to interact: by freely rotating the Bloch sphere using a mouse or other input device and activating a measurement operator which collapses the qubit represented by the Bloch sphere into either a $|0\rangle$ or $|1\rangle$ state. When the measurement occurs, a circuit is constructed which assembles a state identical to that represented by the Bloch sphere and runs this circuit on quantum hardware to figure out how the collapse occurs.

Besides being an effective use of quantum hardware for live settings, Q1Synth is a powerful creative tool, and useful for understanding how measurement affects a quantum system. That said, its utility as a pedagogical and analytical tool is limited in some ways. Firstly, by choosing to use a Bloch sphere as the primary representation of a qubit, it is limited to sonifying only single-qubit systems. While it is possible to have multiple instances of Q1Synth running in parallel, with the current implementation they cannot entangle or otherwise interfere with each other, as might be the case for many composite quantum systems. Secondly, the interaction mechanism of freely rotating a Bloch sphere is misrepresentative of how qubits tend to be interacted with in practice; instead of specifying a quantum state and having a circuit or other transformation system generated after, most quantum systems take the form of applying transformations to one state and getting another as a result[21]. As such, Q1Synth's ability to sonify existing quantum systems is limited.

Our work counters these issues by generalizing to any number of quantum states, whether they are states of a qubit-based system or of any other type of quantum system. Furthermore, our work sonifies existing sequences of quantum state transformations, and so can be applied to nearly any existing quantum system. That said, our mechanism requires a quantum simulation, so there is an upper limit to the size of the system that we can sonify.

5 Limitations and Future Work

5.1 Simulated Quantum Systems

As mentioned, one intrinsic limitation of our current method is that it works only with simulated quantum systems, due to needing to be aware of the probabilities after each step in the circuit. It is possible to work around this by incrementally running each step on quantum hardware immediately followed by a measurement to get an approximation of the probabilities, but this is somewhat computationally expensive and infeasible for real-time applications[9].

5.2 User Interface

A further limitation of our current implementations is the user interfaces. Given the pedagogical bent of our method, ease-of-use is crucial for lowering the barrier of entry for a wider audience[22]. Both Python and Max have sophisticated user interface toolkits, and incorporating these into our current algorithms would be a valuable avenue for future development.

5.3 Entanglement

The technique outlined here is also limited in that it does not explicitly account for entanglement. While we do implicitly sonify entanglement through our definition of the state space, right now there is no difference between the sonification between a 4-state system and an entangled 2x2-state system. While measurement operators would behave differently on both systems, there are no aural cues in our method to make clear that entanglement is happening. Extending our approach to clearly depict entanglement is thusly a worthy goal.

5.4 Phase

Besides entanglement, our system also does not account for phase. Phase is often more difficult to conceptualize than superposition, and thus representing it effectively could have profound pedagogical impacts. One potential sonification method for phase would be to change the starting position in a motif depending on the phase; e.g., if the phase was $\frac{\pi}{2}$, then start the motif from halfway through. This could represent phase well, as it accounts for the negligibility of a global phase factor: if motifs in all instruments are shifted the same amount, then there would be no noticeable difference between them!

6 Conclusion

The superposition sonification method we outline in this paper presents a gamut of advantages over existing superposition sonification methods, while also leaving a high degree of flexibility for implementation. As such, it may provide a solid backbone for more advanced sonifications of quantum systems in the future, and help lead to a broader and more intuitive understanding of quantum superpositions!

6.1 Acknowledgements

We would like to thank Eduardo Reck Miranda for pioneering the field of quantum computer music, and for inviting us to submit a paper to ISQCMC. We would also like to thank Adrian German for creating a space for quantum computing at Indiana University Bloomington, and for his help and support in learning this field. Thanks to John Gibson and Chi Wang at the Center for Electronic and Computer Music for their support as well!

References

[1] Dandison Ukpabi, Heikki Karjaluoto, Astrid Bötticher, Anastasija Nikiforova, Dragoş Petrescu, Paulina Schindler, Visvaldis Valtenbergs, Lennard Lehmann, and Abuzer Yakaryilmaz. Framework for understanding quantum computing use cases from a multidisciplinary perspective and future research directions, 2023.

[2] Andy Matuschak and Michael A. Nielsen. Quantum computing for the very curious, 2019.

[3] Megan Watzke and Molly Porter. New nasa black hole sonifications with a remix. 2022.

[4] Walker Smith. The sound of molecules, 2021.

[5] John Rodgers and Willie Ruff. Kepler's harmony of the world: A realization for the ear: Three and a half centuries after their conception, kepler's data plotting the harmonic movement of the planets have been realized in sound with the help of modern astronomical knowledge and a computer-sound synthesizer. American Scientist, 67(3):286–292, 1979.

[6] Qiskit contributors. Qiskit: An open-source framework for quantum computing, 2023.

[7] MicroQiskit contributors. Microqiskit, 2021.

[8] Max.

[9] John Watrous. Fundamentals of quantum algorithms. Qiskit.

[10] Qiskit Development Team. Statevector, 2023.

[11] Python.

[12] Google colaboratory.

[13] Midi 1.0 detailed specification, 1996.

[14] Apple. Logic pro, 2023.

[15] Bitwig GmbH. Bitwig studio, 2023.

[16] Omar Costa Hamido. QAC: Quantum-computing aided composition. In Quantum Computer Music, pages 159–195. Springer International Publishing, 2022.

[17] Gary Mooney and Harish Vallury. qmuvi, 2023.

[18] Sanjeev R. Kulkarni. The human auditory and visual systems, 2001.

[19] Tamara Smyth. Music 175: Loudness, 2019.

[20] Eduardo Miranda, Peter Thomas, and Paulo Itaboraí. Q1synth: A quantum computer musical instrument. Applied Sciences, 13:2386, 02 2023.

[21] Andy Matuschak and Michael A. Nielsen. Quantum mechanics distilled, 2020.

[22] Behnam Faghih, Mohammad Azadehfar, and Seraj Katebi. User interface design for e-learning software. 01 2014.