Reproducing Xenakis's Stochastic Music: A Modern Python Implementation

Melisa-Maria Mircica
melisa-maria.mircica@s.unibuc.ro

Ionut-Florin Voinea
ionut-florin.voinea@s.unibuc.ro

Abstract

This project addresses the reproducibility challenge of Iannis Xenakis's algorithmic compositions by reconstructing his stochastic music algorithm in modern Python. The original FORTRAN IV code used for works such as ST/10 is difficult to execute and analyze. We translated the core stochastic logic into Python using NumPy, implementing the Exponential Distribution for timing and Square Root transformation for pitch generation. Our approach validates the mathematical foundations described in Formalized Music while making these pioneering techniques accessible through contemporary tools. The implementation successfully generates MIDI outputs that preserve the characteristic asymmetrical, dense textures of Xenakis's stochastic compositions.

1 Introduction

The problem we address is the reproducibility of Iannis Xenakis's algorithmic compositions. The original FORTRAN IV code used for stochastic works such as ST/10 is difficult to execute or analyze in modern computing environments. Our goal is to reconstruct and validate the stochastic algorithm based on mathematical distributions described in Formalized Music and translate it into a modern Python environment.

We started with the principles detailed in the chapter "Free stochastic music from the computer" from the book *Cybernetics, Art and Ideas* and utilized the original FORTRAN IV source code as our precise technical specification. We translated the core stochastic logic—Exponential Distribution for time and Square Root transformation for pitch—into Python using NumPy.

We chose this project because we were fascinated by the direct connection Xenakis established between music and computer science. It was interesting to see how complex art could be created using strict mathematical principles, bridging the gap between algorithmic thinking and creative expression.

Previous research on Xenakis's algorithmic composition has explored both the theoretical foundations and practical implementations. The seminal work *Formalized Music: Thought and Mathematics in Composition* (Xenakis, 1971) provides the mathematical framework underlying his stochastic approach. Recent scholarly work has examined the computational aspects of his compositions, analyzing how probability distributions shape musical structure and timbre selection in works like ST/10 and Achorripsis.

Each team member contributed significantly to different aspects of the project. The first author prepared and formatted the input data for the Atrees composition, extracted all parameters and built the required tables and matrices, created visualizations including a heatmap of the probability matrix E, and helped test and verify the correctness of data and probabilities. The second author recreated the main stochastic algorithm from the original FORTRAN code, implemented the event generation logic in Python, and converted generated events into MIDI files using the pretty_midi library.

Through this project, we learned how to read and parse the Atrees input data from the original FORTRAN code in Python and discovered how structured rules and randomness can work together to create music. In the future, we would like to explore more complex orchestration techniques and investigate how different probability distributions affect the resulting musical character.

2 Approach

The first step was to identify and correctly interpret the two primary inputs required by Xenakis's algorithm. We identified the exact mathematical formulas used to introduce randomness. For ex-

ample, the duration between notes (T) is calculated using the Exponential Distribution, which is key to creating the asymmetrical, dense textures characteristic of Xenakis. After that, we needed the precise numbers that controlled the randomness. Since the raw data was supplied as a long, continuous series of digits, we had to use the rules embedded in the original FORTRAN program to parse the data into distinct blocks. This step was crucial for accurately extracting the Timbre Probability Matrix (instrument chance table) and the Pitch Constraints (HAMIN and HAMAX).

With the inputs defined, we built the simulation engine using Python and NumPy to handle the calculations. The engine operates through a continuous process of generating sound events. The program runs through a master loop that manages musical "sequences." In each sequence, a new density (DA) is determined, and this value constantly affects the generation of tones.

For every note, we applied Exponential Distribution logic to calculate the precise moment of its attack (TA), ensuring the music sounds naturally random and asymmetrical. We calculated the exact probability of choosing an instrument (e.g., glissando vs. pizzicato) by performing linear interpolation on the Timbre Probability Matrix, making the orchestration dynamically dependent on the current DA. Pitch values are generated according to the square root transformation logic embedded in the original FORTRAN code.

Finally, the numerical data produced by the engine is converted into an audible score using the pretty_midi library, allowing the stochastic patterns to be rendered as actual sound.

Figure 1 shows the Timbre Probability Matrix (Matrix E) as a heatmap, visualizing how different instrument classes (Clasa 1-12) are selected based on density levels (Niv. 1-7). The brightest cells indicate higher probabilities—for instance, Class 12 has a 0.45 probability at density level 1, while Class 8 peaks at 0.41 for density level 5. This matrix is central to Xenakis's algorithm, as it controls the orchestration dynamically throughout the composition.

The complete source code and data files are available in our GitHub repository at https://github.com/TeoRoGamingYG/AMI-Iannis-Xenakis. We used Python 3.x with NumPy for numerical computations and pretty_midi for MIDI generation. The processing

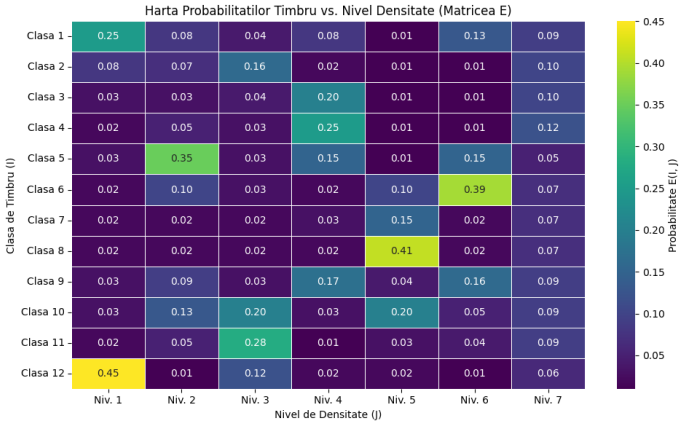

Figure 1: Heatmap of the Timbre Probability Matrix (E) showing the relationship between instrument classes and density levels.

time for generating a complete composition is approximately 2-3 seconds on a standard laptop. No machine learning or deep learning tools were used; instead, we relied on classical probability distributions and deterministic algorithms as specified in Xenakis's original mathematical framework.

3 Conclusions and Future Work

Now that we have completed this project, we recognize that we could have planned the workflow and structure more carefully. Testing ideas incrementally to understand the musical output better would have improved our development process. We could also have documented our interpretation decisions more thoroughly, as some aspects of the FORTRAN code required interpretation.

There are several ways to improve this project. Adding real-time feedback, generating actual audio instead of just MIDI, or creating interactive visualizations of the stochastic processes could make the project more engaging and educational. Implementing additional compositions from Xenakis's catalog would also validate our approach more comprehensively.

We genuinely enjoyed this project—it was engaging to combine programming and music, and it felt creative compared to regular assignments. The challenge of decoding historical computational music techniques was both intellectually stimulating and artistically rewarding.

We learned how algorithmic and stochastic processes can create evolving musical structures and gained practical experience implementing and ex-

perimenting with generative music in Python. Understanding how mathematical abstractions translate into aesthetic experiences deepened our appreciation for both computer science and music composition.

For future projects in this course, we suggest exploring interactive or live generative music systems, other algorithmic composition techniques such as cellular automata or L-systems, or projects that integrate visualization and audio synthesis together. Immediate audio feedback and a more playful exploration of results would make such projects more fun and creatively satisfying.

References

Iannis Xenakis. 1971. *Formalized Music: Thought and Mathematics in Composition*. Indiana University Press.

James Harley. 2004. *Xenakis: His Life in Music*. Routledge.