-Pattern Synthesized Music-

By- Owen Fay, Azalea Houle, Lucas Furlong, Logan Stoddard

Innovation Scholars, Patterns and Symmetry – Spring 2025

University of New Hampshire

-Objective-

The goal of our project is to create an **audial representation** of a randomly-generated pattern, something we learned about prior in the course. Different factors in the visual pattern that influence these sounds are:

- Number of points in the graph
- Symmetries
- Scale of points
- Ripples

By using multiple mathematical techniques, our group was able to accomplish this goal.

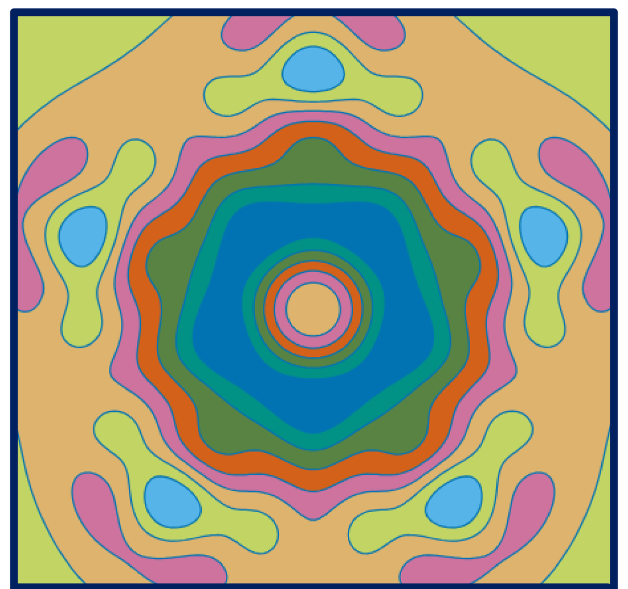

Fig. A
-This is an example of a random pattern generated through Julia, by using different forms of these graphs.

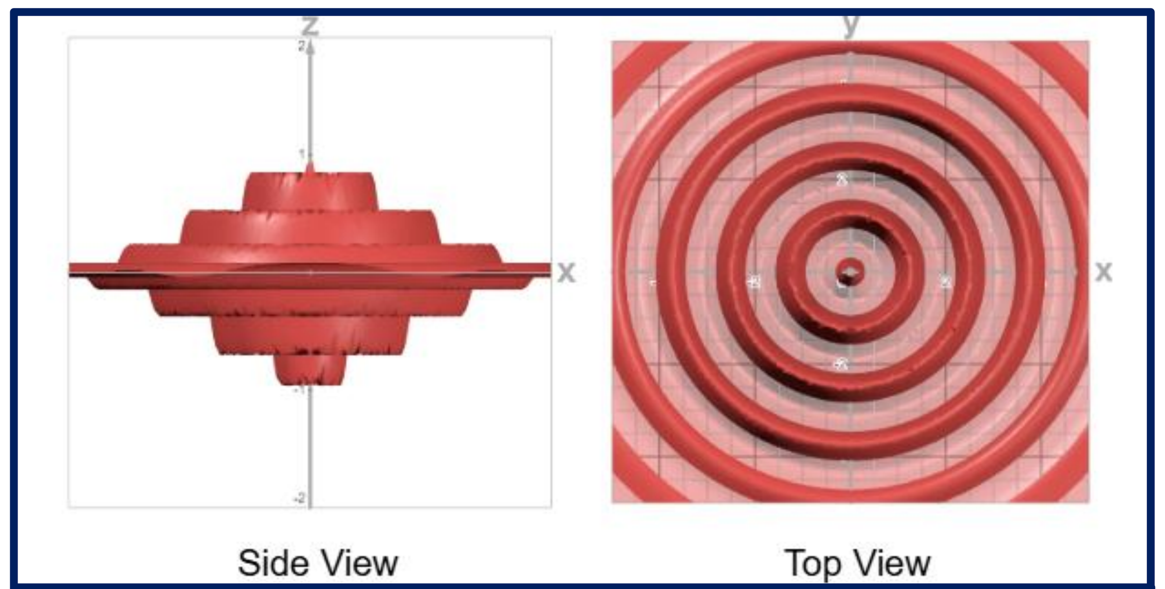

Fig. B
-This is the graph created from rotating the 2D graph around the z axis to create a ripple plot, that will then have a color assigned based on the depth when viewed from the top.

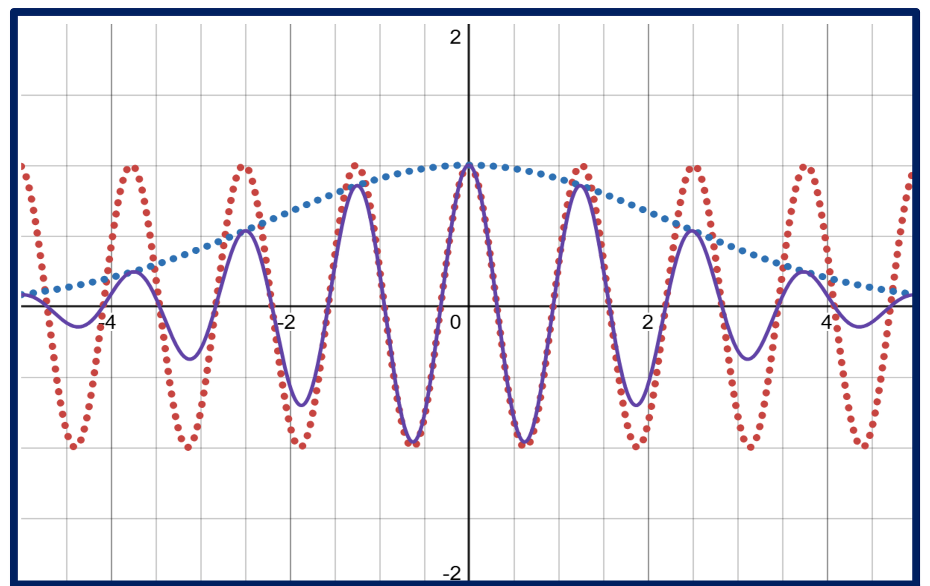

Fig. C
-The patterns are created by first using the base function of $\cos(kx)e^{-ax^2}$. This graph shows $\cos(kx)$ (dotted red), e^{-ax^2} (dotted blue) and $\cos(kx)e^{-ax^2}$ (solid purple).

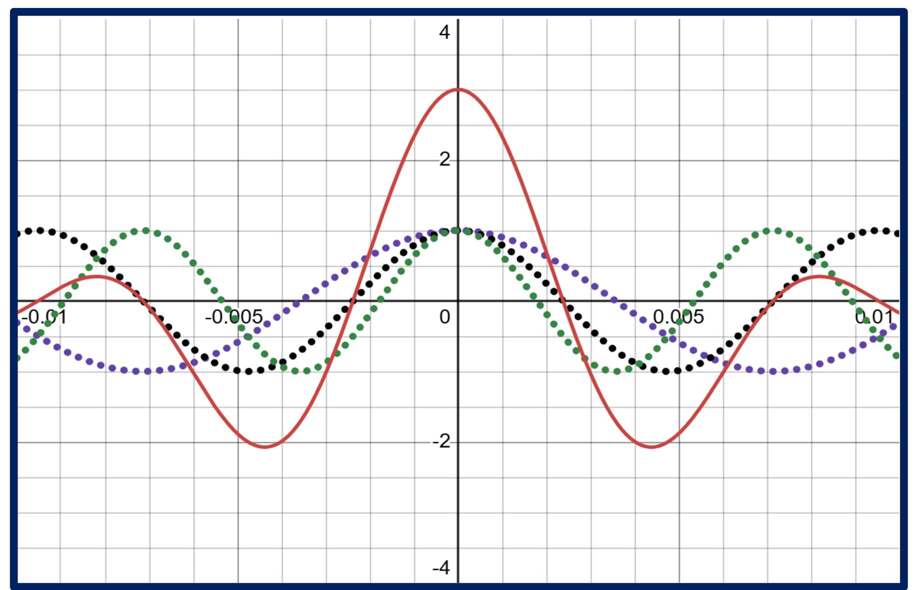

Fig. D
-This graph shows sound waves of the root $\cos(440x)$ (dotted green), the 5th $\cos(660x)$ (dotted black), the octave $\cos(880x)$ (dotted purple), and the wave produced by adding the 3 functions together (solid red).

-Outline-

By taking the original random pattern from Juila, we obtain a graph like the one shown in **Figure D**

- These graphs are obtained by taking a cross-section of the 3-D graph in **Fig. A** (as the patterns are all 3-dimensional)
- This is done by setting x and y to a single line and plotting f(x, y) as a two-dimensional graph

- Using this graph (which looks very sinusoidal), we plot points discretely on it and run these samples through the **Fast Fourier Transform**
- The output of the FFT is what we use for the sounds

-Fast Fourier Transform (FFT)-

- A way of generating sound frequencies from a sinusoidal function
- **Julia**, along with various other programming languages, has the FFT built in as a function
- This function takes in points on a graph and outputs a vector containing multiple values.
  - These values are **magnitudes**, the volumes of their respective **frequencies**, which are pitches
- Part of the algorithm we made **sorts** through the magnitudes, as they are not organized optimally for the sounds to be played **in order of pitch**
- Once these are all sorted by frequency, the sounds are played, layered on top of each other to make a **cluster** of pitches
- These sounds can also be played **staggered** by looping through the frequency/magnitude matrix and playing one sound at a time

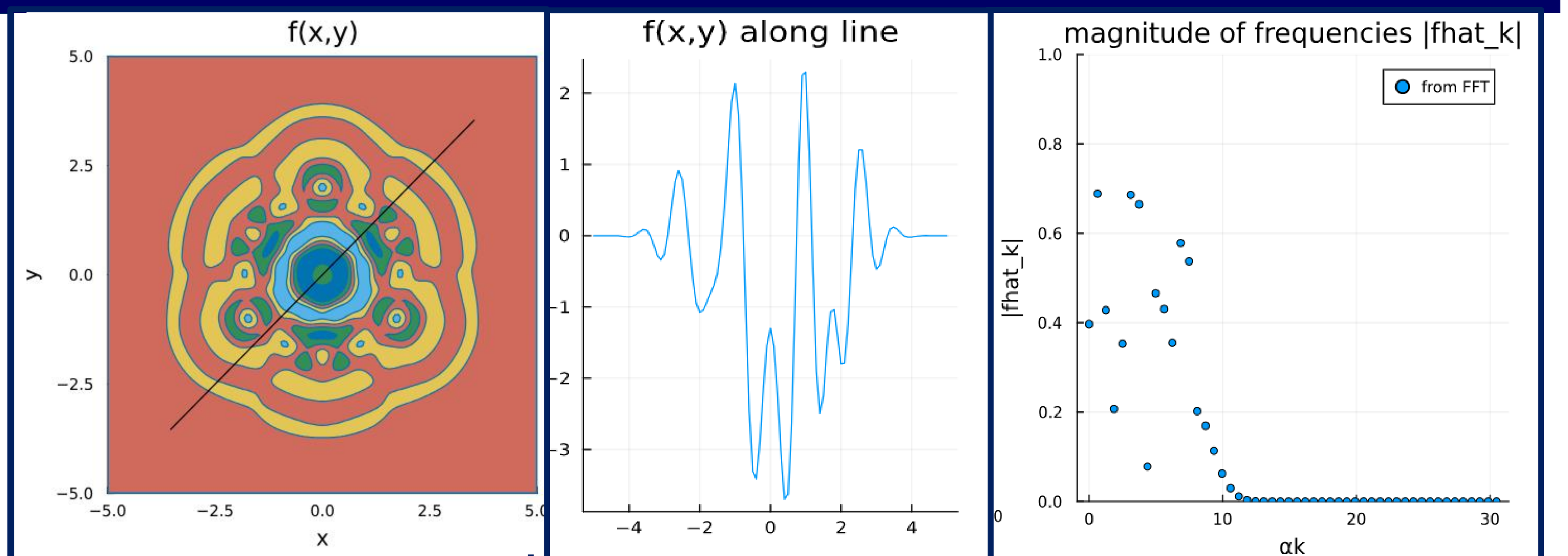

Fig. E
This figure shows 3 graphs, comparing the input of the pattern generated to the **FFT** and to the output. The first graph shows the pattern generated and where the cross-section is taken from. The second graph shows the **cross-section** taken from the pattern, which is a sinusoidal function. The **FFT** takes the cross-section and returns a set of magnitudes; these magnitudes are the volumes of their respective frequencies. The third graph shows the magnitudes of each frequency (note: most are zero). These are the sounds being played.

Credits for the Julia code go to Professor John Gibson, Associate Professor in Integrated Applied Mathematics in the Mathematics and Statistics Department, and Instructor of the Patterns and Symmetry cohort of Innovation Scholars