Scientific Python Interactive Data Acoustic Modeling

<https://github.com/ExclusiveDisjunction/CSPFinal>

Project Report

COP2080

Florida Polytechnic University

Fall 2023

submitted by:

**Triskaidekaphobia**

Hollan Sellars – CS

Zane Wolfe – CS

Emilio Garcia - CS

submitted to:

Professor Christian Navarro

12/10/2023

# Section 1: Introduction

In today’s modern world of the internet, audio and video is everywhere. YouTube, learning languages, even classes can all be online. This means that we should all strive to make sure we have the most intelligible audio to reach as many audiences as possible. In big rooms with many surfaces around, it would be very hard to understand what the speaker is saying, since there are so many reverberations. We seek to write a program that will take in audio files, prerecorded, and determine their RT60 values, as well as determine the most intelligible scenario for the file.

To approach this task, we used the Python programming language, as well as common, open-source libraries to make our various tasks. To keep our program on track, we used task management software and created epics. Our team worked to create our features, and then test them, to keep our program running properly from the start. Our program is capable of properly cleaning & converting files, calculating the frequencies and RT60 value, and then displaying all results to the user. These tasks enable users of our program to ensure that they are clear when they are ready to present what they have worked on!

At the end of the project, we all took away good practices and design patterns to use in other applications. For instance, we each learned how to build good UI layouts using MVC, and then how to handle event occurrences. This is practical for any other program we could make, as it separates big parts of the UI into different chunks that can be more easily changed. Furthermore, we mastered our use of libraries like matplotlib and NumPy, as they were used very frequently in our project to determine RT60 values and frequency ranges. Furthermore, we learned about the components of sound, and how we can make sure that anything that we make in the future is at its peak intelligibility.

# Section 2: Design & Planning

## 2.1 Business Needs

In a finished product, our program is required to input an audio file, from many popular file formats, and then display audio intelligibility information about the file. This means that the program will load that file, ensure it is in proper format, and then preform various calculations (such as frequency analysis) on the file to extract values. One of the main values is the RT60 value, which is to be plotted on its own graph. With all of this in mind, our team devised a plan to break apart the tasks into sections, and then work on each section, testing along the way. This approach allowed us to complete the project in a little less than three weeks’ worth of work. Our project included the usage of basic python utilities (file reading, logging, lists, dicts), as well as using more advanced libraries such as tkinter and pydub.

## 2.2 Design Objectives

In our project, we sought to use any input audio file to preform our calculations. At the end of the process, the program will convert any input file to a .wav and convert any dual channels to a mono channel. This enabled the system to calculate the required values and did not have to worry about left and right channels (which would deeply complicate things). Furthermore, our input files were recorded using our phones, and the test files were provided by our professor.

## 2.3 Functional Requirements

As per instructed, our project was required to use the Model View Controller (MVC) design pattern. This involved splitting our UI files into multiple separate modules that would interact with each other and drive the chain of events. Furthermore, tools such as Configuration and Log were separated into their own modules so that one may simply call upon them with ease instead of having to track them down. These tools enabled all modules to keep track of their processes (for debugging) and allowed for them to share and transmit data using Config.

## 2.4 Limitations

Our project had limiting factors relating to performance. Due to the slower speed of Python, our project would not run as fast as it would in other compiled languages. In addition to this, we had many issues when writing the code of reading the raw buffer from our file. The data was provided using pydub, however, NumPy was required to simplify the array into legible values to perform calculations.

# Section 3: Agile User Stories

## Epic One: User Interface

In this epic, we constructed the roadmap to make the user interface using the MVC pattern. Initially, our project did not use the MVC pattern. This was because we were trying to construct a UI that would simply output our data for testing reasons. This was practical for development but needed to change for the final release. Therefore, we set out our plan to construct the MVC pattern. This went through several steps:

1. Construct Basic UI without MVC.
   1. Tasked to Emilio
   2. Involved the following tasks:
      1. Construct the root and basic grid layout, as well as a title.
      2. Include basic button control to open file.
      3. Display total output graph on UI.
2. Separate UI using MVC pattern.
   1. Tasked to Zane
   2. Involved the following tasks:
      1. Using the basic UI, split the different elements into the MVC design pattern, with each element of the pattern in its own module.
3. Include more graphs into MVC using Config.Configuration
   1. Tasked to Zane & Hollan
   2. Involved the following tasks:
      1. Create Config.Configuration module.
      2. Write UI bins for the new graphs.
      3. Write button that will swap out images.
      4. Write controller functions that will swap out images properly.
4. Test final UI
   1. Tasked to Hollan & Emilio
   2. Involved the following tasks:
      1. Ensure that the file loads upon click.
      2. Ensure that the overall graph loads and opens from the Config.Configuration path.
      3. Ensure that the three different RT60 graphs with different frequencies load with the Config.Configuration paths.
      4. Ensure that the three images display on the UI.
      5. Ensure that all resources get cleaned up when the window is closed.

## Epic Two: Data Analysis

The previous epic would be useless without this one, as it is the one that is crunching the numbers. This epic describes how we will use python to preform all calculations onto the wave files and then create their corresponding graphs.

1. Generate function that will calculate RT60 values.
   1. Tasked to Zane
   2. Involves the following tasks:
      1. Take in the file, and then perform the calculations on it to get the needed values.
      2. Using the data obtained, calculate the RT60.
      3. Plot results, output to path defined in Config.Configuration.
      4. Present results to user.
2. Compute Lowest, Mid, and Highest Resonance Frequencies.
   1. Tasked to Zane
   2. Involves the following tasks:
      1. Take in file, open them and extract information.
      2. Perform required calculations to grab the proper information from the file.
      3. Return results to user.
3. Compute Intelligibility.
   1. Tasked to Hollan
   2. Involves the following tasks:
      1. Gather all frequency values.
      2. Calculate the best frequency for voice intelligibility.
      3. Present results to user.

# Section 4: Results

Our program was recorded in the IST Aula Manga, and in an IST classroom. This was chosen because the Aula Manga has many surfaces to bounce off and causes many reflections and reverberations. In the classroom, however, there is much less surfaces to bounce off, and the audio is more easily understood. Pictures of the two rooms are posted below.

A room with many tables and chairs

Description automatically generated

Figure 1, IST Classroom

Figure 2, Aula Managa Room

## 4.1 Final Product Testing

A large building with stairs and chairs

Description automatically generated with medium confidenceWe tested our project by comparing the results of our program to other students working on this project. This enabled us to see within what range our results were off from it, and if they were off, we corrected our results to work closer to their results. This enabled us to see that we should all be getting close values, so if we are off from the average of everyone, then our results are skewed and our calculations needed to be re-evaluated. Furthermore, with the UI, we simply tested all possible inputs (using other file types, ensuring that results posted in log are visible on UI), and ensured that our resources were properly cleaned up (opened files were closed, ect.).

Our program was able to properly calculate the RT60 and frequency values for input files. This went well due to us doing research on the topic to ensure that we were completing the correct calculations. The program was able to efficiently load the information created by the calculators and meet all requirements.

# Section 5: Conclusions and Project Summary

At the end of our trip, we were able to successfully create our program. Our SPIDAM program is able to properly determine the values for RT60, Low, Mid, and High range frequencies, and then compute the best voice intelligibility. This all together has created a complete functional program that has many use cases.

## 5.1 Individual Contribution Table

|  |  |
| --- | --- |
| Team Member | Contribution |
| Zane Wolfe | 40% |
| Hollan Sellars | 40% |
| Emilio Garcia | 20% |

## 5.2 Individual contribution Reports

Zane Wolfe oversaw the data analysis. Having the best understanding of audio, he wrote the code to calculate the values from the audio files. Once he had calculated the values, he wrote the code to output it all into graphs to be rendered on the UI and worked with the UI’s construction. To add to this, he was tasked with splitting the basic UI into the MVC pattern.

Hollan Sellars was tasked with managing the backend tools, such as logger and configuration, as well as overseeing the UI’s design and layout. He took the information that Zane’s calculations provided and constructed the UI to take in and display this information to the user. Additionally, he wrote the base code that would read and clean files.

Emilio Garcia was mainly tasked with constructing the base UI, and then testing the program. With the basic UI, he made the layout for the first image, title, and information grid. This was later put into use when Zane split the UI into the MVC pattern. Later in the development, he worked to quality control the program and ensure that it works properly, as well as giving the proper results.

## 5.3 Individual Reflection Reports

Hollan

At the end of this project, I was able to wrap up my understanding of common python libraries and working with UI. In the modern world, Python is used extensively as a tool to create more complex things, as it can be easily run and debugged. This is useful when you are trying to work on projects without investing tons of time into the basis of the program. Furthermore, the rich library of modules provided by other developers means that there are many tools already available to do what you want.

Additionally, I learned how to work with MVC design pattern. This pattern is prevalent in any programming language using a UI, as it allows for separation of the individual parts of the UI, making it easier to update and manage. Furthermore, this design pattern allows you to create more complex UI without adding many extra parts; simplifying the process. I can use these design principles to work with many other programs.

Zane

I learned I gained an understanding of sound file encoding and calculations. During this project, I learned how to convert any file into a .wav format. I also learned the mechanics of reverberation and the calculation of rt60 using .wav file data. Furthermore, I learned how to use MVC design effectively.

Additionally, I explored the utilization of pydub for in-depth analysis of .wav data. I also learned how to effectively use a scrum board to efficiently organize and oversee the tasks crucial to completing the project. Lastly, I learned how to create a dynamic UI using tkinter. As well as how to use a cfg file to setup a config.

Emilio-

Overall, this project taught me many things not just regarding the importance of programing but actual skills I need to improve on. When it comes to programming it taught me how to properly structure a GUI and implement certain aspects of functions from other code into that GUI so they can interact with each other and work. Also, when working with a gui and trying to get images to display properly, it showed me how important it is to format a gui so all that needs to be displayed is displayed correctly. It also showed me the importance of cleaning a file because it could lead to problems when trying to work with a file.

This project also taught me I need to work on my communication skills when working with a team. During this project I felt I did not communicate with my team enough and was eventually left behind leading to my minimal contribution to the project. This project also taught me I need to be more forward when collaborating with others because during the project I felt like I couldn’t do anything because I was afraid, I would mess the other’s work up and would cause everything to not work.