Creating an Interface for Quantum Measurements

Xavier Gilbert

July 2024

Mentor Nishaad Khedkar

Thanks to Rob, Chao, Billy, and the rest of RSL!
Problem:
• There are many difficulties involved in quantum device measurement, because the development of these devices is recent, they can often work in unpredictable ways.
• In this case, the ability to quickly and easily visualize data becomes a priority.

Goal:
• Develop a VSCode extension to visualize and interact with superconducting quantum device measurement data.
This image depicts a quantum computing refrigerator, also known as a which is essential for cooling qubits to extremely low temperatures near absolute zero.
Reflection Measurements in the Frequency Domain

Ringdown measurement is a way to measure the quality factors of resonators/cavities.

Cavities have a quality factor (Q), which is defined as the ratio of power stored to power lost.

When energy is introduced into the cavity, a portion of it is absorbed, and the remainder is reflected.

The graph illustrates the energy that is reflected from the cavity.

While the drive is active, the cavity absorbs more energy until an equilibrium is reached between the absorbed and reflected energy. When the drive is deactivated, the energy left within the cavity begins to decay.
Description:
• The extension is designed to streamline the visualization and interaction with measurement data within the Visual Studio Code environment.

Features:
• Displays graphs of quantum device measurements.
• Allows users to click on plots to view and manipulate the corresponding data.
Technical Details

Languages:
• Typescript: Chosen for its strong typing, which helps catch errors early and improves code quality. Offers modern JavaScript features with type safety, making development more efficient.
• Python: Scripts for processing measurement data.
• HTML: Displays measurement data.

Framework: VSCode API
• Provides a robust set of tools for building extensions, including commands, views, and webviews. Ensures seamless integration with the VSCode environment, leveraging its built-in functionalities.
To activate, type command into the VSCode command palette.
Next, select the directory containing your data folders.
The extension automatically displays the images from the chosen directory. Each image has an associated JSON file containing data.
Clicking on an image brings up an “Image Detail” window. This window displays an enlarged version of the image along with some of the data associated with it.
Finally, a Python interactive window is opened. Upon activation this window creates a variable using the parent directory of the image and its data. It then imports two Python modules essential for data analysis. Subsequently, a class is instantiated, providing functionality to manipulate the data as required.
Future Direction and Features

- Added customizability
- Adaptive analysis dropdown
- Interactive graphs
- Quality of life upgrades (keyboard shortcuts)
- Expanded use cases

Prototype of Adaptive analysis dropdown.
Thank you!