Project Briefing: A Hybrid Quantum-Classical Approach to Portfolio Optimization

1. Introduction: The Goal and The Problem

Project Goal: To develop a scalable and efficient method for selecting an optimal portfolio of 10 bonds from a universe of 31 available assets.

The Core Problem: Combinatorial Explosion Portfolio optimization is a classic NP-hard problem. With 31 assets, the number of possible 10-bond portfolios is over 66 million. Checking every single combination is computationally infeasible for classical computers. Our initial goal was to frame this business problem in a way that could be solved by optimization algorithms, specifically those suited for quantum computers.

---

2. The Initial Approach: Direct QUBO for 15 Assets

Our first attempt was a standard and academically sound approach using a **QUBO (Quadratic Unconstrained Binary Optimization)** model.

- **How it Works:** We represented each asset with a binary variable (if selected, if not). We then translated our financial objectives into a single mathematical cost function that a solver can minimize.
    - **Risk Targeting:** A penalty was added for deviating from the target risk (spreadDur) within defined credit quality buckets.
    - **Portfolio Size:** A penalty was added if the final portfolio did not contain exactly 10 bonds.
- **The Quantum Solver (QAOA):** We used the **Quantum Approximate Optimization Algorithm (QAOA)**, a leading candidate for solving such problems on quantum hardware.
- **The Result:** For a small subset of **15 assets**, this method worked perfectly on a local machine. It successfully found an optimal portfolio, demonstrating that the QUBO formulation was correct.

---

3. The Scaling Challenge: Why the Initial Approach Failed for 31 Assets

When we attempted to scale the direct QUBO method from 15 to 31 assets, we immediately encountered a critical roadblock: **the local machine's kernel crashed.**

- **Computational Cost:** The size and complexity of the QUBO matrix grow quadratically with the number of assets. A 31x31 QUBO problem requires simulating a 31-qubit quantum system with dense interactions. This level of complexity exhausted the memory and processing power of a standard local machine.

- **The Takeaway:** A direct, brute-force application of quantum algorithms is not yet feasible for realistically sized problems on near-term hardware or simulators. **A more intelligent strategy was required.**

---

4. Our Solution: A Hybrid "Divide and Conquer" Workflow

To overcome the scaling wall, we designed a hybrid quantum-classical strategy that breaks the large, intractable problem into smaller, solvable pieces. This workflow leverages the strengths of both classical and quantum computation.

Step 1: Divide & Cluster (Classical Machine Learning) The first step is to use a classical computer to intelligently partition the problem.

- **Method:** We use the **K-Means clustering algorithm** from the Scikit-learn library.
- **Features:** The algorithm groups the 31 assets into 5 distinct clusters based on their core financial characteristics: **risk (spreadDur) and return (oas)**.
- **Outcome:** Instead of one massive 31-asset problem, we now have five small, independent sub-problems (each with 5-9 assets). This is the "divide" phase.

Step 2: Conquer (Quantum-Inspired Optimization) Next, we solve each of these small sub-problems using our quantum algorithm.

- **Method:** For each cluster, we build a miniature QUBO model with a local objective (e.g., "find the best 2 assets *from this group*").
- **Solver:** We use the **QAOA algorithm** (via the OpenQAOA library) to find the "champion" assets within each cluster.
- **Why it Works:** Simulating a small 5- to 9-qubit system is computationally trivial for a local machine. This allows us to use the quantum algorithm's power to explore complex correlations within each small group without crashing the system. This is the "conquer" phase.

Step 3: Combine (Classical Post-processing) Finally, we use a classical computer to assemble the final portfolio from the results of the quantum runs.

- **Method:** We collect all the "champion" assets identified by QAOA into a single, elite candidate pool (e.g., 11-15 assets).
- **Final Selection:** A simple classical method (like sorting by the best return-to-risk ratio) is used to select the final 10-bond portfolio from this highly qualified pool.

- **Outcome:** A final, optimized 10-bond portfolio that is globally coherent and built from locally optimized components.

---

5. Code Workflow Summary

The implementation is broken into a series of sequential scripts:

1. 01_classical_clustering.py: Loads the 31 assets and uses K-Means to partition them into 5 clusters.
2. 03_generate_all_qubos.py: Automates the process of creating a unique QUBO model for each of the 5 clusters.
3. 05_solve_all_clusters_local.py: The quantum step. It loops through each cluster's QUBO and solves it locally using OpenQAOA.
4. 06_assemble_and_visualize.py: Collects the quantum results, builds the final portfolio, and generates the risk-return visualization.

---

6. Conclusion: A Practical Path to Quantum Advantage

This project successfully demonstrates a practical and scalable method for applying quantum-inspired optimization to real-world financial problems.

- **We proved that a direct approach is not yet feasible for realistically sized problems on current hardware.**
- **We designed a hybrid workflow that successfully circumvents these limitations.**
- **By combining classical machine learning for problem decomposition and quantum algorithms for targeted optimization, we were able to solve the full 31-asset problem efficiently on a local machine.**

This hybrid strategy represents a powerful template for tackling large-scale combinatorial optimization challenges and provides a clear roadmap for leveraging the power of quantum computing today.