Strategic Roadmap

Arife Nur Ayaz

November 2025

The core of this strategy is to integrate four key upgrades identified from our research: the ADAPT-QAOA engine , hard-constraint XY-mixers , a Warm-Starting (WS-QAOA) procedure , and a noise-resilient CVaR objective function.

Phase 1: Foundations & Classical Benchmarking

Week 1: Project Scaffolding & Data Pipeline

- Implement the data-fetching pipeline to retrieve historical price data.
- Write and validate the classical pre-processing functions to calculate expected returns (μ) and the covariance matrix (Σ).

Week 2: Classical Solvers & Advanced QUBO Formulations

- Implement a classical solver (e.g., using CPLEX or Gurobi) for the baseline MVO-QUBO. This is the primary benchmark we must beat.
- Expand the problem formulation module to include:
  - Mean-Absolute Deviation (MAD): A Linear Programming (LP) based model.
  - CVaR-Risk Model: A downside-risk formulation.
  - Transaction Costs: A non-convex formulation.

Phase 2: Core Quantum Engine Development

Week 3: Constraint-Aware Mixers (XY-Mixers)

- Implement the standard QAOA X mixer (soft-constraint).
- Implement the hard-constraint XY-Mixer (specifically the QAMPA variant) to enforce the fixed-budget constraint $\sum z_i = B$ by operating only within the feasible subspace.

Week 4: ADAPT-QAOA Algorithmic Engine

- Implement the baseline Standard QAOA algorithm (fixed ansatz)
- Implement the advanced ADAPT-QAOA algorithm, which iteratively grows a problem-tailored ansatz.

Week 5: Engine Integration & Initial Test

- Integrate the XY-Mixer as the operator pool for the ADAPT-QAOA engine, creating a constraint-aware, adaptive algorithm.
- Preliminary Benchmark (Simulator): ADAPT-QAOA (XY-Pool) vs. Standard QAOA (X-Mixer).
- Criteria: CNOT count, optimization parameters, and layers (p) to convergence.

Phase 3: NISQ Performance Enhancements

Week 6: Warm-Starting (WS-QAOA)

- Implement the "Warm-Start" procedure.
  - Solving the classical relaxation (QP or SDP) to find c^*.
  - Preparing the initial state

Week 7: CVaR Objective & Post-Processing

- Replace the standard expectation value objective $\langle H \rangle$ with the noise-resilient CVaR objective for the classical optimizer.
- Implement the classical local-search post-processing step to "polish" the final sampled bitstring.

Phase 4: Full-Scale Benchmarking & Analysis

Week 8: Full Module Testing (MVO)

- Run a comprehensive benchmark on noisy simulators for the MVO problem.
- Compare:
  - Standard QAOA (Cold-Start, $\langle H \rangle$ Objective)
  - Upgraded Module: ADAPT-QAOA (Warm-Start, CVaR Objective, XY-Mixer, Local Search).
  - Classical MIP Solver (Optimal Baseline).

Week 9: Advanced Problem Testing (MAD & Transaction Costs)

- Rerun the benchmark on the harder, non-convex problems (MAD-QUBO, MVO w/ transaction costs) where classical solvers struggle.
- Analyze and synthesize all results, preparing for the final deliverables.

Phase 5: Final Deliverables

Week 10: Documentation & Dissemination

- Finalize all code, write comprehensive documentation, and clean the GitHub repository.
- Create the interactive Jupyter Notebook tutorial, guiding a user through the advanced end-to-end module (WS-ADAPT-QAOA)
- Write the blog post for the Qiskit community, explaining the project, the baseline, and why the advanced, noise-resilient components (ADAPT, WS, CVaR) are essential for scalable quantum optimization.

Deliverables

- Primary Deliverable 1: Public GitHub repository with all documented code.
- Primary Deliverable 2: Interactive Jupyter Notebook tutorial.
- Primary Deliverable 3: Qiskit community blog post.