Project Proposal: Comparative Analysis of Quantum–Classical Hybrid Hedge Fund Architectures

1. Overview

This project aims to design, implement, and evaluate a **quantum–classical hybrid hedge fund framework** where different algorithmic tools — both classical and quantum — are tested under diverse market regimes. The core idea is to identify **which risk, optimization, and forecasting tools perform best under specific market conditions** and **how large language models (LLMs) select and combine these tools** when acting as the system's agentic decision-makers.

Ultimately, this will lead to:

- A **comparative dataset** linking market conditions   tool performance   LLM decision logic.
- A **research benchmark** for hybrid quant systems integrating AI and quantum computation in real financial decision loops.

---

2. Objectives

1. **Comparative study of tools:** Evaluate multiple *risk assessment*, *portfolio optimization*, and *strategy selection* algorithms — both classical and quantum — across different historical and simulated market regimes.
2. **LLM-driven tool selection:** Use an **agentic LLM controller** (LangGraph or Autogen) to dynamically select which tool(s) to deploy given the current regime (bull, bear, neutral, volatile).
3. **Performance and interpretability analysis:** Quantify not only returns but *why* certain tools were chosen by the LLM, creating transparent mappings between market states, algorithmic decisions, and outcomes.
4. **Comprehensive logging and reproducibility:** All data flows, decisions, and model calls will be fully logged and versioned (MLflow + TimescaleDB + vector store) to support research reproducibility and future extension.

---

3. System Architecture

3.1 Core Modules

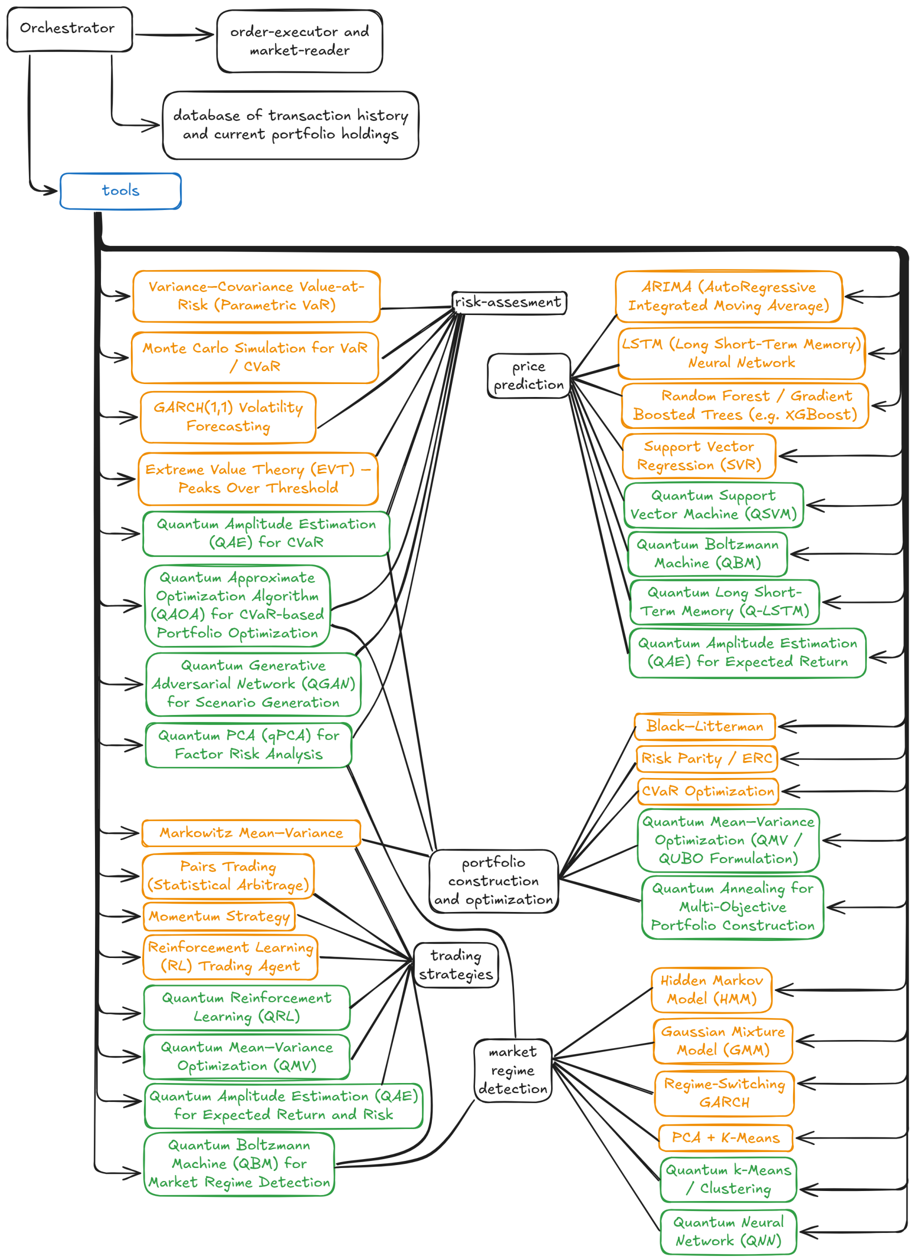

Figure 1: architecture

Module	Function	Tools / Frameworks
Market Data Ingestion	Collect & preprocess market and macro data	`yfinance`, `Polygon.io`, `TimescaleDB`, `Prefect`
Risk Assessment	Compute VaR, CVaR, GARCH volatility, QAE-based risk	`PyPortfolioOpt`, `arch`, `scipy`, `Qiskit Finance`
Price Forecasting	Predict returns using classical ML and quantum ML	`LSTM`, `XGBoost`, `QSVM`, `QGAN`, `PennyLane`
Portfolio Construction & Optimization	Optimize portfolios under multiple objectives	`cvxpy`, `PyPortfolioOpt`, `QAOA`, `QMV`, `D-Wave Ocean SDK`
Market Regime Detection	Identify market states	`HMM`, `GMM`, `RS-GARCH`, `qPCA`, `QBM`
Trading Strategy Module	Simulate & evaluate momentum, mean-reversion, RL trading	`vectorbt`, `FinRL`, `StableBaselines3`, `QRL (PennyLane)`
Agentic LLM Orchestrator	Decide which tools to invoke under current regime	`LangGraph`, `OpenAI GPT-5 API`, `tool registry`
Logging & Benchmarking Layer	Record results, tool choices, and metrics	`MLflow`, `TimescaleDB`, `Grafana`, `Prometheus`

---

4. Experimental Design

1. **Dataset selection:** Use 10–20 years of equity & ETF data (daily & intraday granularity) to capture multiple market cycles.
2. **Regime labeling:** Segment history into regimes via volatility clustering (RS-GARCH/HMM) and validate with macro indicators (VIX, yield curve, GDP growth).
3. **Tool evaluation loop:**
    - For each time segment, run all applicable algorithms (classical & quantum).
    - Measure: Sharpe ratio, Sortino, drawdown, VaR/CVaR, hit ratio, volatility, stability.
    - Record which tool performs best under each regime.
4. **LLM-driven selection:**
    - The LLM agent receives recent market features + performance logs.

- It chooses the next best combination of tools (risk + optimizer + predictor).
- Log both *choice rationale* (LLM reasoning trace) and *resulting performance.*

5. **Comparative metrics:** Build a multi-dimensional table linking:

$$(\text{Market Regime}) \to (\text{Tool Selection}) \to (\text{Performance Metrics})$$

enabling analysis of tool–regime–LLM decision alignment.

---

5. Tools, Frameworks, and Stack

Layer	Tools / Frameworks
Core Language	Python 3.11
Computation & Orchestration	Prefect, Ray
Databases	Postgres + TimescaleDB
ML / DL Frameworks	PyTorch, scikit-learn, XGBoost
Quantum Frameworks	Qiskit, PennyLane, D-Wave Ocean
Agentic / LLM Orchestration	LangGraph, Autogen, OpenAI GPT-5 API
Monitoring & Logging	MLflow, Grafana, Prometheus, Loki
Backtesting / Simulation	vectorbt, FinRL
Version Control & CI/CD	GitHub + Actions, Docker Compose

---

6. Expected Outcomes

- **Empirical maps** showing which algorithmic families perform best under which market conditions.
- **Quantitative benchmarks** of quantum speed-quality tradeoffs versus classical baselines.
- **LLM selection logs** revealing interpretability of AI reasoning in financial contexts.
- **Open-source dataset** of tool-performance-regime mappings for academic use.
- **2–3 research papers** (Springer, IEEE, or Quantitative Finance) detailing performance analysis, hybrid architecture, and AI-based decision orchestration.

---

7. Timeline (Approx. 20 Weeks)

Phase	Duration	Focus
Setup & Data Backbone	2 weeks	Ingestion, DB setup, logging layer
Classical Pipeline	4 weeks	VaR/CVaR, Markowitz, ML predictors
Quantum Extensions	5 weeks	QAE, QAOA, QBM, QSVM integration
LLM Agent & Decision Layer	3 weeks	Tool selection logic, LangGraph orchestration
Comparative Study & Benchmarking	4 weeks	Market regime testing, backtesting, metrics
Reporting & Publication	2 weeks	Documentation, research draft

---

8. Research Contribution

This project creates a **new hybrid research benchmark** at the intersection of:

- **Computational Finance** (quant risk & optimization)
- **Quantum Computing** (QAE, QAOA, QBM)
- **Agentic AI Systems** (LLM tool orchestration)

It aims to demonstrate how **AI agents select analytical tools** under varying regimes, quantify **cross-domain performance patterns**, and propose **quantum–classical collaboration strategies** for future financial systems.