EVOQUANTLAB: TRAINING-FREE DUAL-LOOP EVOLUTION FOR AUDITABLE QUANTITATIVE RESEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Automated alpha factor discovery is critical for modern quantitative research. While large language models (LLMs) have increasingly been integrated into quantitative workflows, existing agent-based approaches remain highly susceptible to severe backtest overfitting. To address this challenge, we propose EVOQUANTLAB, a training-free, dual-loop self-evolving framework for robust factor discovery. Instead of relying entirely on LLM generation, EVOQUANTLAB deploys an inner genetic-programming (GP) loop that dramatically improves token efficiency while evolving typed factor expressions under structural and diversity constraints. An outer diagnosis loop then systematically audits evidence across cycles to drive protocol-level improvements. Furthermore, our evaluation pipeline applies multi-metric validation to rigorously filter out spurious factors before deployment. Extensive out-of-sample experiments across both daily and hourly frequencies—covering equities and cryptocurrency—demonstrate the robustness of EVOQUANTLAB. Specifically, on daily CSI 500 and S&P 500 tasks, EVOQUANTLAB achieves leading annualized returns of 18.34% and 21.30%, consistently outperforming existing agent baselines. On high-turnover hourly cryptocurrency data, EVOQUANTLAB delivers a 37.91% annualized return while effectively avoiding execution collapse.

1 INTRODUCTION

Automated alpha discovery has become an increasingly important direction in quantitative research (Xiao et al., 2024; Lopez-Lira & Tang, 2026). Recent large language model (LLM) agents can formulate economic hypotheses, generate factor implementations, interpret backtest feedback, and iteratively refine candidate signals, enabling substantially larger research spaces than traditional manual workflows (Li et al., 2025; Tang et al., 2025; Han et al., 2026; Wang et al., 2026). Systems such as R&D-Agent-Quant, AlphaAgent, QuantaAlpha, and FactorMiner demonstrate that LLMs can automate significant portions of factor and model discovery across heterogeneous financial data.

However, scaling automated research also amplifies a fundamental problem: *backtest overfitting*. An agent may repeatedly generate and evaluate large numbers of factors, transformations, windows, and model variants, making it increasingly likely that apparently strong signals arise from noise or adaptive selection rather than persistent predictive structure (Harvey et al., 2016; Bailey & López de Prado, 2014). This issue is particularly severe for iterative LLM agents, since development results are repeatedly fed back into subsequent generations. Existing systems mainly focus on improving candidate generation, memory, reflection, or S&Psearch (Yang et al., 2024), but stronger search alone can further increase the effective number of trials (Kou et al., 2024). Meanwhile, relying on LLM calls for every candidate is unnecessarily expensive when many factor variations are symbolic transformations that can be explored more efficiently (Shi et al., 2025; Romera-Paredes et al., 2023).

To address these challenges, we introduce EVOQUANTLAB, a training-free dual-loop self-evolving framework for robust alpha discovery. At the inner level, EVOQUANTLAB combines LLM-generated economic hypotheses with genetic programming (GP), which efficiently evolves typed factor expressions through selection, crossover, and mutation while maintaining structural and population diversity. At the outer level, the framework audits completed research cycles for evidence of overfitting, instability, excessive complexity, and selection bias, and uses these diagnoses to improve the research protocol for subsequent cycles. Importantly, protocol changes only affect future cycles rather than retroactively modifying completed evaluations. Together with a multi-metric validation pipeline that

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

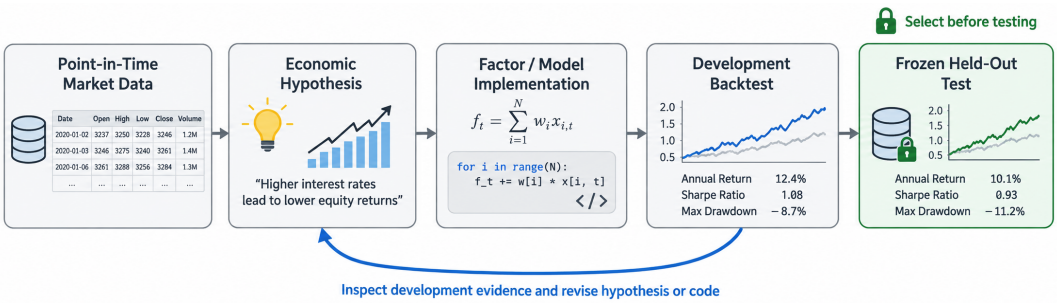

Figure 1: A conventional quantitative-research workflow. Iteration over development backtests is necessary, but repeated inspection makes the complete search history part of the statistical experiment. The held-out test must remain outside this feedback path.

filters unstable, redundant, and economically inconsistent candidates, this design allows the search process itself to evolve while preserving disciplined out-of-sample evaluation.

We evaluate EvoQuantLab across daily and hourly quantitative prediction tasks covering equities and cryptocurrency. On CSI 500 and S&P 500, EvoQuantLab achieves annualized returns of 18.34% and 21.30%, respectively, compared with 14.02% and 15.23% from the strongest baselines. On hourly cryptocurrency data, it achieves a 37.91% annualized return despite the shorter horizon and substantially higher turnover. Beyond final performance, our experiments show that GP provides an efficient mechanism for large-scale factor exploration, while outer-cycle diagnosis improves robustness across successive research cycles.

Our contributions are:

- We introduce EvoQuantLab, a training-free dual-loop framework that jointly evolves factor expressions and the quantitative research process to address adaptive backtest overfitting.
- We combine LLM-guided hypothesis generation with diversity-preserving genetic programming, enabling efficient exploration of large symbolic factor spaces with substantially reduced reliance on repeated LLM inference.
- We develop a diagnosis-driven outer evolution mechanism and rigorous multi-metric validation pipeline, and demonstrate strong out-of-sample performance across daily and hourly equity and cryptocurrency tasks.

2 Related Work

Automated alpha factor discovery. Formulaic alpha discovery seeks interpretable expressions that predict future asset returns. Classical approaches rely on human-designed signals or search over symbolic expressions, with genetic programming (GP) providing a natural mechanism for evolving executable factor trees through crossover and mutation (Koza, 1992; Kronberger et al., 2026). Evolutionary search has been applied to trading-rule discovery (Neely et al., 1997; Allen & Karjalainen, 1999), while AutoAlpha extends this paradigm with hierarchical evolutionary search for formulaic alphas (Zhang et al., 2020). Learning-based approaches instead optimize the generation process directly; for example, deep symbolic regression frameworks (Petersen et al., 2019)and AlphaGen (Yu et al., 2023) formulate formulaic alpha discovery as reinforcement learning. Despite their effectiveness, these methods typically operate within a predefined search space whose operators and search procedure remain fixed. EvoQuantLabretains an explicit GP search engine but allows validated research knowledge and the search vocabulary to evolve across cycles.

LLM agents for quantitative research. Recent work uses large language models to automate broader parts of quantitative research, connecting hypothesis generation, factor implementation, empirical evaluation, and iterative refinement(Wang et al., 2024). R&D-Agent(Q) coordinates research and development agents for joint factor–model optimization (Yang et al., 2024; Zhang et al., 2024); AlphaAgent combines LLM-driven factor generation with regularization against redundancy, complexity, and alpha decay (Tang et al., 2025); and systems such as QuantaAlpha and FactorMiner further explore iterative factor discovery with evolutionary search and accumulated research experi-

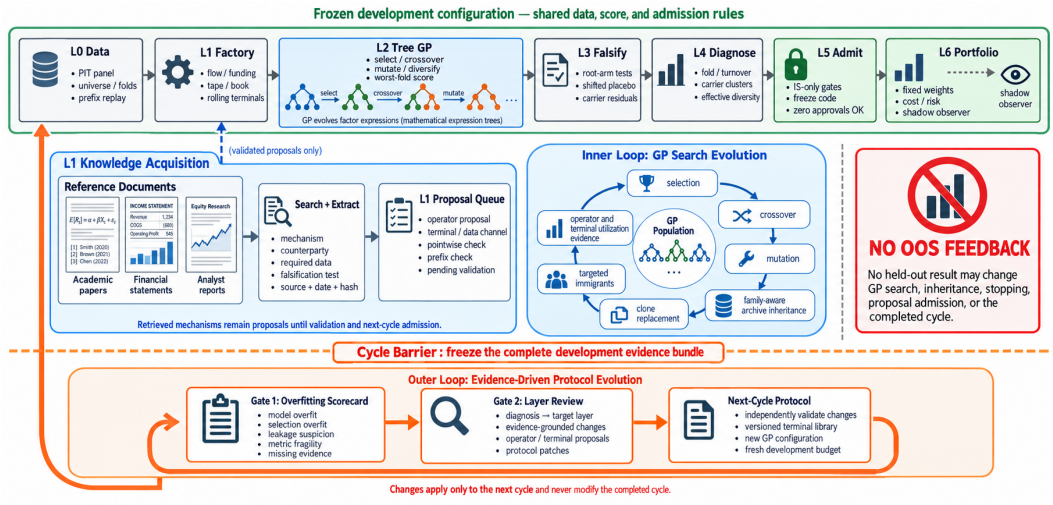

Figure 2: Overview of EVOQUANTLAB. A frozen L0–L6 evidence pipeline evaluates every candidate under a fixed data and evaluation contract. The fast inner loop evolves factor expressions and reallocates search capacity using development-only evidence, while the slow outer loop diagnoses overfitting after each completed research cycle and proposes a versioned protocol for the next cycle. Information from held-out evaluation never writes back into the current cycle.

ence (Han et al., 2026; Wang et al., 2026; Li et al., 2023). These systems demonstrate the potential of LLMs as autonomous quantitative researchers, but also expose the model repeatedly to noisy backtest feedback in open-ended trial-and-error loops(Lu et al., 2024) In contrast, EVOQUANTLABseparates semantic research evolution from candidate-level optimization: the LLM updates reusable research knowledge, while a deterministic, budget-accounted GP engine performs factor search.

Self-evolving agents. Beyond finance, self-evolving agents aim to improve their own reasoning procedures, workflows, or architectures through accumulated execution feedback and experiential memory(Shinn et al., 2023; Zhao et al., 2024). Pioneering efforts search over executable programs and algorithmic heuristics using LLMs in an evolutionary loop(Romera-Paredes et al., 2024; Wang et al., 2023). ADAS automatically discovers code-defined agent designs(Hu et al., 2025), while AFlow searches over code-represented agentic workflows using iterative execution feedback(Zhang et al., 2025). AgentSquare similarly evolves and recombines modular components such as planning, tool use, and memory(Shang et al., 2025). This line of work shows that adaptation can occur at the system level rather than solely through model parameters. EVOQUANTLAB brings this perspective to quantitative research, but evolves research knowledge and symbolic search across cycles under a frozen evidence protocol, preventing held-out outcomes from influencing subsequent evolution.

3 EVOQUANTLAB

Automated quantitative discovery faces two intertwined challenges: efficiently exploring vast symbolic spaces, and preventing adaptive backtest overfitting caused by repeated experimentation. EVOQUANTLAB addresses this by decoupling exploration across two distinct timescales: an *inner fast loop* evolving factor expressions via genetic programming (GP) within a frozen protocol, and an *outer slow loop* auditing research evidence across cycles to update protocol rules.

3.1 FORMAL SETUP & IMMUTABLE TASK CONTRACT

Let $\mathcal{D} = (\mathcal{D}^{\text{train}}, \mathcal{D}^{\text{dev}}, \mathcal{D}^{\text{oos}})$ be chronologically ordered market data. A research cycle c is governed by a frozen protocol $\Pi_c = \langle \mathcal{S}_c, \mathcal{T}_c, \mathcal{O}_c, \mathcal{M}_c, \mathcal{B}_c \rangle$, where $\mathcal{S}_c$ specifies the task and data contract, $\mathcal{T}_c$ denotes the active terminal set, $\mathcal{O}_c$ the typed operator set, $\mathcal{M}_c$ the evaluation rules, and $\mathcal{B}_c$ the search and trial budgets. To eliminate lookahead bias, Π_c enforces point-in-time availability and *prefix replay invariance*: for any $\tau < \tau'$, features computed on $\mathcal{D}_{1:\tau}$ must agree with those computed on $\mathcal{D}_{1:\tau'}$ over the shared prefix. Within cycle c, EVOQUANTLAB maintains a population of typed factor trees $\mathcal{P}_c = \{g_1, \ldots, g_K\}$, where K denotes the population size and each candidate $g \in \mathcal{G}(\mathcal{T}_c, \mathcal{O}_c)$ belongs

to the valid expression space induced by the active terminal and operator sets. When evaluated on an asset–time observation, g produces a scalar factor score. Candidates are evaluated across F development folds via a robust worst-fold objective penalized by structural tree complexity:

$$f_{\text{dev}}(g; \Pi_c) = \min_{1 \le k \le F} \text{Metric}_k(g) - \lambda \, \text{Complexity}(g), \tag{1}$$

where $\text{Metric}_k(g)$ denotes the task-specific utility on development fold k, defined such that larger values indicate better performance; $\text{Complexity}(g)$ is a pre-registered scalar complexity penalty derived from tree size and depth, and $\lambda > 0$ controls its strength. Equation 1 operates strictly on $\mathcal{D}^{\text{dev}}$; the holdout split $\mathcal{D}^{\text{oos}}$ is physically inaccessible during search. Training data $\mathcal{D}^{\text{train}}$ are used only for task-specific parameter estimation when required (e.g., fitting normalization or baselines), whereas candidate selection and protocol evolution are driven entirely by development evidence.

3.2 The Frozen L0–L6 Evidence Pipeline

Candidates are sequentially validated through a cycle-frozen seven-layer pipeline (Fig. 2), advancing only upon passing preceding gates. The pipeline begins at **L0 (Data)**, which assembles canonical panels, tradable universes, and purged fold masks under strict prefix invariance. Next, **L1 (Factory)** instantiates typed, economically grounded terminals spanning order flow, funding, microstructure, and momentum, while **L2 (Tree GP)** searches typed symbolic expression trees guided by the worst-fold fitness objective in Equation 1.

Surviving expressions undergo economic falsification at **L3 (Falsify)** through only-arm/drop-arm ablations, sign-flip and participant-swap tests, and cross-sectional residualization against pre-registered factor carriers. A candidate survives only if its hypothesized core mechanism remains directionally consistent and the full expression provides reproducible incremental value. At each timestamp, we cross-sectionally residualize candidate scores against a pre-registered carrier matrix and re-evaluate the residual signal across development folds. Candidates whose predictive power vanishes after residualization are rejected as redundant with known factor families.

Subsequently, **L4 (Stress Battery)** stress-tests signal stability across regime splits (market trend $\times$ volatility), execution delay curves, asset clusters, and matched-budget block permutations. Candidates that clear all development hurdles reach **L5 (Admit)**, where their implementation, terminal manifest, data version, and metric configuration are frozen and recorded by content hashes before OOS access. Finally, **L6 (Portfolio)** maps admitted signals into trading strategies under fixed volatility targeting, risk constraints, and transaction costs.

3.3 Loop A: Inner GP Search & Knowledge Ingestion

L1 Knowledge Ingestion. Instead of expanding features blindly, L1 extracts economic mechanisms from academic papers, financial disclosures, and analyst reports into structured *Reference Cards*. These proposals enter an isolated *L1 Proposal Queue*. A candidate primitive must clear both a pointwise reference check and a prefix causality test before becoming eligible for admission across cycle boundaries. Each active terminal $j \in \mathcal{T}_c$ maintains a development evidence tuple $e_j = (u_j, s_j, r_j, d_j)$, tracking utilization u_j, candidate survival s_j, residual contribution r_j, and cross-fold stability d_j. Primitives with weak standalone power but sustained interaction gains are retained as *interaction-only* to preserve diversity.

Tree-Based Factor Evolution. Within cycle c, population $\mathcal{P}_c$ evolves across generations via tournament selection. Offspring are generated via typed subtree crossover and point mutation over valid grammar nodes. Syntactic duplicates and expressions with pairwise correlation $\rho > 0.95$ are replaced with targeted immigrants from underrepresented families. Elites are preserved in a family-stratified archive instead of a monolithic top-K list, guaranteeing structural diversity within the trial budget.

3.4 Loop B: Diagnosis-Driven Protocol Evolution

The outer loop activates only after an entire cycle crosses the **Cycle Barrier** (Fig. 2). Unlike the inner GP loop, Loop B does not modify candidates within the current cycle. Instead, it audits accumulated training and development evidence and proposes bounded protocol changes that take effect only in the next cycle. Importantly, both diagnosis and protocol evolution are performed without access to the held-out test set.

Gate 1: ML Overfitting Scorecard. Gate 1 audits execution traces for evidence of adaptive overfitting, including train-to-dev degradation, cross-seed variance, tree complexity, concentration, label sensitivity, and search multiplicity. To control for selection multiplicity, let $m_t^{\text{real}} = f_{\text{dev}}(g_t; \Pi_c)$ denote the selection score of the t-th evaluated candidate under protocol Π_c, and define $m^{\text{real}}(T) = \max_{1 \le t \le T} m_t^{\text{real}}$ as the best score achieved across T real search trials. We run B_0 independent null pseudo-searches with the same trial budget under a pre-registered block-permutation null, and record $m_b^{\text{null}}(T) = \max_{1 \le t \le T} m_{b,t}^{\text{null}}$.

$$\hat{p}_{\text{sel}}(T) = \frac{1 + \sum_{b=1}^{B_0} \mathbf{1}\big[m_b^{\text{null}}(T) \ge m^{\text{real}}(T)\big]}{B_0 + 1}. \tag{2}$$

Equation 2 evaluates whether the search procedure itself yields alpha beyond a budget-matched random search over noise. Gate 1 outputs a diagnostic status $\mathcal{V}_{\text{diag}} \in \{\texttt{Pass}, \texttt{Overfit-Model}, \texttt{Overfit-Selection}, \texttt{Suspect-Leakage}, \texttt{Metric-Fragile}, \texttt{Insufficient}\}$.

Gate 2: Routed Review & Child Protocol Synthesis.

Gate 2 uses $\mathcal{V}_{\text{diag}}$ to route the review toward the protocol components most likely responsible for the observed failure mode. For example, suspected leakage triggers an audit of L0 data availability and split construction; model overfitting routes to complexity control; selection overfitting routes to the search space and trial budget; and metric fragility routes to robustness analysis of the evaluation criteria. The reviewer then proposes a bounded patch set $\mathcal{A}_c$, whose changes are applied only to the next protocol version:

$$\Pi_{c+1} = \text{Apply}(\Pi_c, \mathcal{A}_c), \qquad h_{c+1} = \text{Hash}(\Pi_{c+1}). \tag{3}$$

Completed results from cycle c remain permanently associated with Π_c; the patch set never retroactively modifies candidate scores, selection decisions, or evaluation rules from the completed cycle. Appendix D provides case studies illustrating how EvoQuantLab redirects unproductive search, rejects deceptively strong candidates, and validates model-suggested knowledge before admission.

4 Main Experiments

We evaluate whether dual evolution improves out-of-sample performance under a reproducible evaluation protocol. We benchmark EvoQuantLab against four representative agent-based alpha discovery baselines: **AlphaAgent** (Tang et al., 2025), which regularizes LLM factor exploration against redundancy and signal decay; **QuantaAlpha** (Han et al., 2026), an evolutionary framework optimizing formulaic alphas through LLM-driven operators; **FactorMiner** (Wang et al., 2026), a self-evolving agent guided by accumulated skill and experience memory; and **RD-Agent-Quant** (Li et al., 2025), a multi-agent framework coordinating data-centric factor mining and model optimization. For a fair comparison, all methods execute their native workflows using the identical foundation model (Claude Opus 5) while sharing the exact same point-in-time data, temporal splits, transaction-cost assumptions, and evaluation suite. The out-of-sample (OOS) data remain strictly frozen and unobserved during the factor discovery process, accessed solely during final validation.

4.1 Tasks and metrics

We evaluate all methods on three quantitative prediction tasks spanning diverse markets, asset classes, and temporal resolutions, as summarized in Table 1. The daily-bar benchmarks cover CSI 500 (China equities) and S&P 500 (U.S. equities) for next-day cross-sectional return prediction. To examine generalization under higher turnover and noise, we evaluate on an hourly cryptocurrency benchmark for next-hour return prediction.

Table 1: Summary of the three evaluation tasks.

Dataset	Frequency	Market	Train / Dev Period	Prediction target
CSI 500	Daily	China equities	2025-03-31–2026-04-30	Next-day cross-sectional return
S&P 500	Daily	U.S. equities	2025-03-31–2026-04-30	Next-day cross-sectional return
Crypto	Hourly	Digital assets	2025-04-01–2026-03-31	Next-hour cross-sectional return

We assess performance along two dimensions: factor predictive ability and realized portfolio quality. Predictive power is captured by the information coefficient (IC, cross-sectional Pearson correlation)

and its temporal stability (ICIR). Portfolio performance is evaluated via an execution-aware long-short strategy with transaction costs, measured by annualized return (AR), cumulative return (Ret.), information ratio (IR), and maximum drawdown (MDD). Full trading and execution details are deferred to Appendix C.

4.2 Daily and hourly results

Table 2 presents the daily-bar results on CSI 500 and S&P 500. On both equity markets, EvoQuantLab consistently outperforms baseline agent frameworks across predictive and risk-adjusted metrics. Specifically, on CSI 500, EvoQuantLab achieves the highest IC (0.0281), ICIR (0.1211), IR (1.613), and cumulative return (5.31%), with a competitive MDD (-4.78%). On S&P 500, EvoQuantLab delivers top-tier performance with an IC of 0.0233, an IR of 1.614, and an annualized return of 21.30%, while maintaining a well-controlled drawdown (-5.82%). Figure 3 depicts the corresponding cumulative excess-return trajectories.

Table 2: Main experimental results on CSI 500 and S&P 500. **Bold** indicates the best result.

	CSI 500 (2026-05-06–2026-08-17)						**S&P 500** (2026-05-01–2026-08-14)					
Method	IC ↑	ICIR ↑	IR ↑	AR ↑	Ret. ↑	MDD ↓	IC ↑	ICIR ↑	IR ↑	AR ↑	Ret. ↑	MDD ↓
AlphaAgent	0.0177	0.1072	1.340	14.02%	4.06%	-7.09%	0.0070	0.0573	1.490	13.67%	3.96%	**-3.48%**
QuantaAlpha	0.0139	0.0893	0.939	9.41%	2.73%	-9.88%	0.0095	0.0663	0.796	7.59%	2.20%	-5.92%
FactorMiner	0.0187	0.0746	1.047	4.11%	1.19%	**-3.26%**	0.0173	0.0954	1.582	15.23%	4.41%	-4.18%
RD-Agent	0.0080	0.0438	0.546	6.31%	1.83%	-7.36%	0.0136	0.0580	0.902	13.07%	3.79%	-7.76%
EvoQuantLab	**0.0281**	**0.1211**	**1.613**	**18.34%**	**5.31%**	-4.78%	**0.0233**	**0.1029**	**1.614**	**21.30%**	**6.17%**	-5.82%

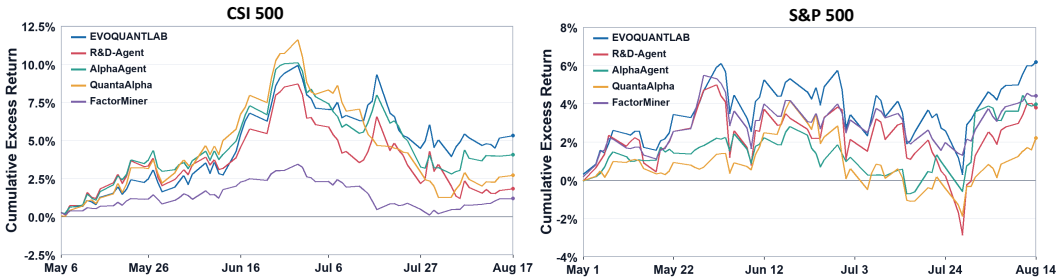

Figure 3: Cumulative excess returns of different approaches on CSI 500 and S&P 500.

	Crypto (2026-04-01–2026-06-09)					
Method	IC ↑	ICIR ↑	IR ↑	AR ↑	Ret. ↑	MDD ↓
AlphaAgent	0.0366	0.1278	-2.385	-27.50%	-5.20%	-12.40%
QuantaAlpha	0.0320	0.1472	-1.665	-10.66%	-2.02%	-4.55%
FactorMiner	0.0086	0.0482	0.952	9.53%	1.80%	-3.48%
RD-Agent	**0.0444**	**0.1807**	-1.870	-16.15%	-3.05%	-8.29%
EvoQuantLab	0.0089	0.0299	**3.709**	**37.91%**	**7.17%**	**-3.23%**

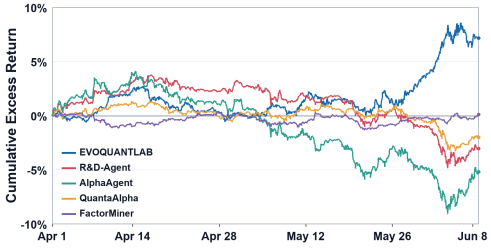

Figure 4: Performance comparison on Crypto. Left: predictive and portfolio metrics of each method. Right: cumulative excess-return trajectories over the OOS period.

The higher-frequency Crypto benchmark reveals a stark divergence between predictive correlation and execution profitability. As shown in Figure 4, several agent baselines generate high predictive correlations on paper but collapse during execution: RD-Agent achieves the highest IC (0.0444) and ICIR (0.1807), yet yields a negative IR (-1.870) and substantial drawdowns (-8.29%). In contrast, EvoQuantLab successfully converts discovered signals into robust trading returns, achieving the highest IR (3.709), cumulative return (7.17%), and the lowest drawdown (-3.23%). This highlights EvoQuantLab's ability to discover executable, noise-resilient alphas in high-turnover regimes.

4.3 Dev-to-OOS Generalization and Overfitting Audit

A central challenge in automated alpha discovery is backtest overfitting, where repeated model inspection and candidate selection produce deceptive in-sample performance that collapses out-of-sample. To directly evaluate whether discovered signals exhibit true out-of-sample persistence, we

perform a pre-registered transfer audit. For each method, we identify the single best-performing factor strictly using development data (Dev IR), freeze its implementation, and evaluate the *identical* factor on the held-out OOS period across all three markets without any re-tuning.

As reported in Table 3, EVOQUANTLAB achieves remarkably consistent positive out-of-sample performance across all three markets, delivering an average OOS IR of 2.521 and preserving positive alpha ($+ \rightarrow +$ transition) in 100% of the tasks. In contrast, existing agent-based baselines exhibit substantial performance degradation: RD-Agent, AlphaAgent, and QuantaAlpha suffer severe IR collapses and sign reversals on both equity and high-frequency crypto benchmarks, reflecting that iterative LLM feedback inadvertently optimizes against development noise. While FactorMiner maintains positive returns, it retains only a single candidate factor per market ($M = 1$), completely bypassing the multiple-testing selection problem.

Table 3: Out-of-sample IR of the single development-selected factor. Means reflect equal weighting across markets.

Method	Crypto	CSI 500	S&P 500	Mean
EVOQUANTLAB	**4.055**	**1.041**	**2.466**	**2.521**
RD-Agent-Quant	-0.229	1.002	-0.082	0.230
AlphaAgent	-2.385	0.798	-0.490	-0.692
QuantaAlpha	-4.999	-0.079	0.819	-1.420
FactorMiner†	0.952	0.547	1.382	0.960

†FactorMiner retains only 1 candidate per market, bypassing candidate selection.

Furthermore, as detailed in the comprehensive audit in Appendix E, EVOQUANTLAB achieves an average positive-sign survival rate of 85.5% across all retained candidates (reaching 95.5% on CSI 500). This confirms that our outer-loop diagnostic gating and frozen evidence pipeline effectively suppress selection bias and prevent spurious signals from surviving to deployment.

5 ABLATION STUDIES

5.1 COMPONENT ABLATION STUDY

Table 4 reports the performance of key architectural components under a strictly matched candidate-evaluation budget. *Cycle 0 only* terminates after the initial iteration, isolating the cumulative benefits of cross-cycle inheritance and evidence-guided refinement. *Cycle 0 without GP updates* ablates iterative genetic search (selection, crossover, and mutation) in favor of uniform random sampling; this yields no qualified factors that satisfy our multi-stage pipeline filters, underscoring that directed evolutionary optimization is essential to factor viability. *No Loop B* retains multi-cycle execution but removes the outer diagnosis loop, preventing protocol-level feedback and evidence-driven search adjustments across cycles. *No L1 search* fixes the terminal and operator spaces to their initial states, quantifying the marginal gain of dynamic operator retrieval across cycles. Finally, *No diversity controller* removes family-stratified inheritance, clone deduplication, and targeted immigrant injection, directly measuring their impact against premature population collapse.

Across all variants, all metrics reflect frozen out-of-sample evaluations. These results highlight two key insights: while directed GP updates are strictly necessary to generate valid, filter-passing alphas, dynamic L1 expansion, Loop B outer diagnosis, and family diversity control are critical drivers for maximizing downstream predictive and risk-adjusted alpha quality.

Table 4: Component ablation on the dual-loop framework. Metrics report out-of-sample performance of the top L5-approved factor chosen via a predeclared selection rule. Values marked with "–" indicate that no sampled candidates passed the validation filters to reach L5 approval.

Variant	IC ↑	ICIR ↑	IR ↑	AR ↑	Ret. ↑
Full EVOQUANTLAB	**0.0089**	**0.0299**	**3.709**	**37.91%**	**7.17%**
Cycle 0 only	0.0067	0.0343	1.570	19.93%	3.77%
Cycle 0 w/o GP updates	–	–	–	–	–
No Loop B	0.0055	0.0281	0.752	6.58%	1.54%
No L1 search	0.0039	0.0120	2.331	27.44%	5.19%
No diversity controller	-0.0029	-0.0156	1.712	10.51%	2.58%

5.2 EVOLUTION ACROSS CYCLES

We further trace how the system evolves across outer cycles rather than comparing only the initial and final states. Figure 5 tracks a representative Crypto run, depicting key search events and the

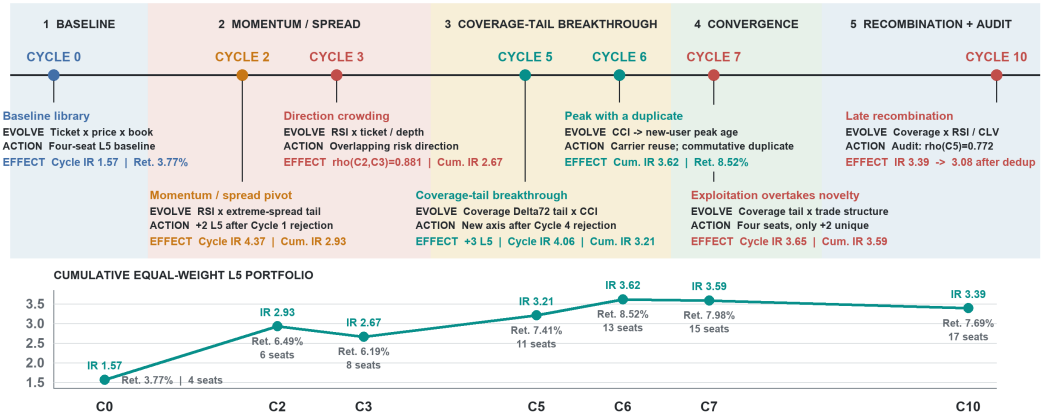

Figure 5: Retrospective trajectory of discovered factor families and the cumulative L5 portfolio across outer cycles on Crypto. Crucially, all OOS IR and Return values shown are computed strictly post-hoc for retrospective analysis; no OOS signal is exposed to the search agent.

cumulative equal-weight portfolio of L5-approved factors to contrast genuine search-space expansion with redundant family refinement.

The trajectory exhibits a clear transition from exploration to exploitation. Initialized with four factors at Cycle 0 (IR 1.57), early evolution targets momentum- and spread-based structures, elevating the cumulative IR to 2.93 by Cycle 2. While Cycle 3 finds extra candidates, their highly overlapping risk profile ($\rho = 0.881$) yields negligible gain, showing that factor quantity alone cannot bolster the portfolio.

A larger improvement occurs in the middle of the run when evolution expands into a previously underexplored factor family. After the rejected Cycle 4 proposals, Cycle 5 introduces coverage-tail signals combined with CCI and contributes three new L5-approved factors, increasing the cumulative IR from 2.67 to 3.21. Continued refinement of this family reaches the peak portfolio IR of 3.62 at Cycle 6, together with an arithmetic return of 8.52%. These transitions suggest that the main benefit of iterative evolution comes from discovering new predictive axes rather than merely recombining candidates within an already saturated region of the search space.

Later cycles show diminishing returns and increasing redundancy. By Cycle 7, exploitation begins to dominate novelty: four portfolio seats are proposed but only two correspond to unique additions, while the cumulative IR slightly decreases from 3.62 to 3.59. The final recombination stage exhibits a similar pattern. Although Cycle 10 continues to construct candidates from coverage, RSI, and CLV signals, the audit identifies substantial overlap with earlier discoveries; deduplication reduces the corresponding IR from 3.39 to 3.08. Overall, the trajectory shows that the evolution process is most effective while it continues to open new factor families, whereas later cycles increasingly revisit correlated structures and approach a performance plateau.

5.3 Evolution Efficiency Analysis

We evaluate the execution and resource efficiency of EvoQuantLab alongside four representative agent baselines: R&D-Agent-Quant, AlphaAgent, QuantaAlpha, and FactorMiner. All methods are evaluated under identical task settings, environment constraints, and hardware allocations. Table 5 details the end-to-end computational and financial footprint across wall time, responsiveness, LLM token and API invocation budgets, and factor discovery throughput.

Time and Search Depth Trade-offs. As reported in Table 5, lightweight baselines such as FactorMiner and R&D-Agent-Quant terminate rapidly (under 25 minutes) and quickly produce initial candidate factors (under 2 minutes), yielding higher nominal throughput (up to 156 L5 factors per hour). However, this rapid completion stems from constrained search horizons and relatively shallow validation cycles. In contrast, EvoQuantLab sustains an extended, deep-horizon exploration trajectory spanning 22h08m, with the first valid candidate materializing at 2h45m. This reflects the thoroughness of EvoQuantLab's iterative refinement and multi-stage evaluation pipeline, prioritizing deliberate, high-conviction factor evolution over superficial high-volume sampling.

Table 5: Efficiency and resource consumption comparison across factor discovery methods. API costs are computed using a standardized pricing schedule frozen across all runs.

Method	Wall time	Time to first valid	Input tokens	Output tokens	LLM calls	API cost	L5 factors / hour
R&D-Agent-Quant	18m39s	1m47s	91745	18007	26	0.909	45
AlphaAgent	25m39s	5m39s	149870	67721	23	2.442	9.36
QuantaAlpha	2h4m	2m29s	767897	455379	94	15.224	10.2
FactorMiner	24m36s	50s	102771	11040	21	0.790	156
EvoQuantLab	22h08m	2h45m	520780	256000	50	9.000	0.95

Token and Financial Budget Economy. Despite running substantially longer than iterative competitors, EvoQuantLab maintains a moderate computational and financial footprint. Specifically, EvoQuantLab incurs only 50 LLM calls with 520,780 input and 256,000 output tokens, translating to an API cost of $9.000. In comparison, QuantaAlpha incurs over 767K input tokens, 455K output tokens, and 94 API calls over its 2-hour run, escalating costs to $15.224 without engaging in multi-round hypothesis evolution of comparable depth. This demonstrates that EvoQuantLab achieves its exhaustive search regime primarily through rigorous numerical and symbolic validation rather than wasteful LLM token churn.

5.4 Backbone Model Analysis

Backbone Comparisons. To assess sensitivity to the underlying backbone, we benchmark Claude Opus5, Claude Sonnet5, and the open-weight Qwen3.8-27B (Fig.6). Claude Opus5 achieves the strongest portfolio-level performance, particularly in risk-adjusted returns and proposal admission, reflecting better structural adherence and generalization. Claude Sonnet 5 delivers competitive intermediate performance across dimensions. Qwen3.8-27B, despite achieving the highest standalone IC, exhibits a lower admission rate and a larger gap between predictive strength and downstream portfolio performance. This pattern suggests that smaller backbones are more prone to overfitting during factor discovery, producing signals that appear strong under localized predictive metrics but generalize less reliably through the full validation pipeline. Nevertheless, its strong raw IC highlights the potential of open-weight models for cost-efficient, locally deployed alpha discovery.

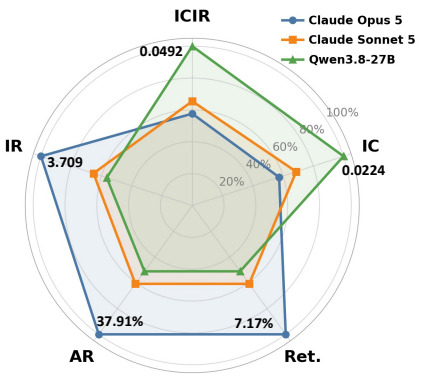

Figure 6: Comparison of EvoQuantLab using different models (11 cycles each). Axes are normalized, with outer points indicating better performance.

6 Conclusion

In this paper, we introduced EvoQuantLab, a training-free, dual-loop self-evolving framework designed to tackle the critical challenge of backtest overfitting and trial multiplicity in automated quantitative research. By decoupling token-efficient symbolic exploration in an inner genetic-programming loop from evidence-driven protocol evolution in an outer diagnostic loop, EvoQuantLab establishes an auditable, causally principled quantitative research workflow under strict immutable task contracts and one-way OOS governance.

Comprehensive evaluations across daily equities (CSI 500 and S&P 500) and high-frequency digital assets demonstrate that EvoQuantLab significantly outperforms representative agent-based baselines, consistently translating predictive correlation into superior realized, risk-adjusted returns. The ablation studies and longitudinal cycle analyses confirm that directed genetic evolution, dynamic knowledge acquisition, and multi-metric falsification are indispensable for preventing population collapse and weeding out spurious signals. Remarkably, the framework achieves these robustness gains while maintaining a modest LLM token and financial footprint. We believe this paradigm of disciplined, multi-timescale self-evolution offers a robust, reproducible foundation for next-generation autonomous scientific and quantitative discovery.

Reproducibility statement

To support independent reproduction, we will release the source code, dependency specifications, experiment configurations, and task adapters for the CSI 500, S&P 500, and hourly cryptocurrency benchmarks. The release will specify the exact temporal splits, point-in-time universe construction, preprocessing and label definitions, transaction-cost assumptions, random seeds, and evaluation procedures. When source licenses prevent direct redistribution of market data, we will instead provide data-acquisition and preparation scripts together with schemas and content hashes needed to reconstruct and verify the experimental inputs. Each run records versioned task and protocol manifests, candidate hashes, append-only trial and LLM-call ledgers, cached predictions, and atomic checkpoints. We will also release anonymized search trajectories and frozen OOS evaluation artifacts so that the reported results and the separation between development and held-out evaluation can be audited.

AI use statement

We used an AI assistant for grammar checking and translation during manuscript preparation. For grammar checking, we used the assistant to identify grammatical errors and suggest corrections to sentence structure, tense, and word usage. For translation, we used it to translate text, with attention to fluency, consistent technical terminology, and preservation of the original meaning.

References

Franklin Allen and Risto Karjalainen. Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, 51(2):245–271, 1999. doi: 10.1016/S0304-405X(98)00052-X.

David H. Bailey and Marcos López de Prado. The deflated sharpe ratio: Correcting for selection bias, backtest overfitting, and non-normality. *The Journal of Portfolio Management*, 40(5):94–107, 2014. doi: 10.3905/jpm.2014.40.5.094.

Jun Han, Shuo Zhang, Wei Li, Yifan Dong, Tu Hu, Yumo Zhu, Xiaomin Yu, Xin Guo, Zhaowei Liu, Kunyi Wang, et al. QuantaAlpha: An evolutionary framework for LLM-driven alpha mining. *arXiv preprint arXiv:2602.07085*, 2026. URL https://arxiv.org/abs/2602.07085.

Campbell R. Harvey, Yan Liu, and Heqing Zhu. ...and the cross-section of expected returns. *The Review of Financial Studies*, 29(1):5–68, 2016. doi: 10.1093/rfs/hhv059.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems, 2025. URL https://arxiv.org/abs/2408.08435.

Zhizhuo Kou, Holam Yu, Junyu Luo, Jingshu Peng, Xujia Li, Chengzhong Liu, Juntao Dai, Lei Chen, Sirui Han, and Yike Guo. Automate strategy finding with llm in quant investment. *seed*, 1:3, 2024.

John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

Gabriel Kronberger, Fabricio Olivetti de Franca, Deaglan J. Bartlett, Harry Desmond, and Pedro G. Ferreira. Guiding multi-objective genetic programming with description length improves symbolic regression solutions, 2026. URL https://arxiv.org/abs/2605.22374.

Yang Li, Yangyang Yu, Haohang Li, Zhi Chen, and Khaldoun Khashanah. Tradinggpt: Multi-agent system with layered memory and distinct characters for enhanced financial trading performance, 2023. URL https://arxiv.org/abs/2309.03736.

Yuante Li, Xu Yang, Xiao Yang, Xisen Wang, Weiqing Liu, and Jiang Bian. R&D-Agent-Quant: A multi-agent framework for data-centric factors and model joint optimization. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Alejandro Lopez-Lira and Yuehua Tang. Can chatgpt forecast stock price movements? return predictability and large language models. *Journal of Financial Economics*, 184:104335, 2026.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024.

Christopher Neely, Paul Weller, and Rob Dittmar. Is technical analysis in the foreign exchange market profitable? a genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32 (4):405–426, 1997. doi: 10.2307/2331231.

Brenden K Petersen, Mikel Landajuela, T Nathan Mundhenk, Claudio P Santiago, Soo K Kim, and Joanne T Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. *arXiv preprint arXiv:1912.04871*, 2019.

Bernardino Romera-Paredes, Mohammad Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan Ellenberg, Peng Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2023. doi: 10.1038/s41586-023-06924-6. URL https://doi.org/10.1038/s41586-023-06924-6.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://nature.com.

Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. Agentsquare: Automatic llm agent search in modular design space, 2025. URL https://arxiv.org/abs/2410.06153.

Yu Shi, Yitong Duan, and Jian Li. Navigating the alpha jungle: An llm-powered mcts framework for formulaic factor mining, 2025. URL https://arxiv.org/abs/2505.11122.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL https://arxiv.org/abs/2303.11366.

Ziyi Tang, Zechuan Chen, Jiarui Yang, Jiayao Mai, Yongsen Zheng, Keze Wang, Jinrui Chen, and Liang Lin. AlphaAgent: LLM-driven alpha mining with regularized exploration to counteract alpha decay. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025. URL https://arxiv.org/abs/2502.16789.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Meiyun Wang, Kiyoshi Izumi, and Hiroki Sakaji. LLMFactor: Extracting profitable factors through prompts for explainable stock movement prediction. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3120–3131, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.185. URL https://aclanthology.org/2024.findings-acl.185/.

Yanlong Wang, Jian Xu, Hongkang Zhang, Shao-Lun Huang, Danny Dongning Sun, and Xiao-Ping Zhang. FactorMiner: A self-evolving agent with skills and experience memory for financial alpha discovery. *arXiv preprint arXiv:2602.14670*, 2026. URL https://arxiv.org/abs/2602.14670.

Yijia Xiao, Edward Sun, Di Luo, and Wei Wang. Tradingagents: Multi-agents llm financial trading framework. *arXiv preprint arXiv:2412.20138*, 2024.

Hongyang Yang, Boyu Zhang, Neng Wang, Cheng Guo, Xiaoli Zhang, Likun Lin, Junlin Wang, Tianyu Zhou, Mao Guan, Runjia Zhang, and Christina Dan Wang. Finrobot: An open-source ai agent platform for financial applications using large language models, 2024. URL https://arxiv.org/abs/2405.14767.

Shuo Yu, Hongyan Xue, Xiang Ao, Feiyang Pan, Jia He, Dandan Tu, and Qing He. Generating synergistic formulaic alpha collections via reinforcement learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023. doi: 10.1145/3580305.3599831. URL https://arxiv.org/abs/2306.12964.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. Aflow: Automating agentic workflow generation, 2025. URL https://arxiv.org/abs/2410.10762.

Tianping Zhang, Yuanqi Li, Yifei Jin, and Jian Li. AutoAlpha: An efficient hierarchical evolutionary algorithm for mining alpha factors in quantitative investment. *arXiv preprint arXiv:2002.08245*, 2020. URL https://arxiv.org/abs/2002.08245.

Wentao Zhang, Lingxuan Zhao, Haochong Xia, Shuo Sun, Jiaze Sun, Molei Qin, Xinyi Li, Yuqing Zhao, Yilei Zhao, Xinyu Cai, Longtao Zheng, Xinrun Wang, and Bo An. A multimodal foundation agent for financial trading: Tool-augmented, diversified, and generalist, 2024. URL https://arxiv.org/abs/2402.18485.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners, 2024. URL https://arxiv.org/abs/2308.10144.

A Dual-Loop Execution Procedure

For completeness, the outer-cycle procedure is given below. All state transitions are deterministic except candidate and patch proposals made by the frozen API model.

1. Load protocol Π_c and verify its content hash. Instantiate the task adapter from the immutable task specification.
2. Run the unchanged L0–L6 GP search engine on train/dev data. Freeze the complete cycle artifact bundle. Abort outer review if the inner run is incomplete.
3. Construct per-terminal evidence from use, survival, ablation, residual, fold, and redundancy artifacts. Loop A emits lifecycle and budget decisions. Optional reference search emits proposal cards only.
4. Construct a `CandidateAuditInput` for every finalist. Gate 1 emits an `MLOverfitReport` and a priority-ordered list of target layers.
5. Gate 2 reviews layer evidence in that order. It emits typed local patches with evidence paths, expected benefit, risks, and a next-cycle validation plan.
6. Reject patches that touch an immutable field, lack evidence, lack a regression test, or attempt to change the completed cycle. Apply admitted patches to create Π_{c+1} and compute its hash.
7. Atomically write the Loop A review, Gate 1 reports, Gate 2 review, next manifest, and outer checkpoint. Resume only from the last atomic checkpoint.

B Gate 1 Evidence Schema

The reference implementation uses conservative defaults only for contract tests. The protocol manifest records every calibrated value, its calibration data, and the commit preceding all OOS reads.

C Evaluation Details

C.1 Metrics

We adopt two classes of metrics to evaluate quantitative performance: factor-level predictive metrics and strategy-level portfolio returns.

Information Coefficient (IC). IC measures the cross-sectional correlation between the model's predicted scores and the subsequent realized asset returns. Following standard practice in quantitative finance, it is defined as:

$$\mathrm{IC}_t = \frac{(\hat{y}_t - \mathbb{E}[\hat{y}_t])^\top (y_t - \mathbb{E}[y_t])}{\sigma(\hat{y}_t) \cdot \sigma(y_t)} \tag{4}$$

Table 6: Minimum Gate 1 evidence. Thresholds are calibrated per task before OOS access.

Dimension	Required fields	Typical red flag
Generalization	train, inner validation, outer development in one metric	sign reversal or small outer/train ratio
Fold/seed stability	worst fold, dispersion, same-sign rate, rank consistency	winner identity changes with seed
Complexity	feature/parameter count, simple-model score, one-SE loss	simpler model is statistically tied
Selection	total and family trials, matched-budget permutation winners	real winner is ordinary under null search
Concentration	date/entity contribution and leave-group-out results	a few dates or assets explain the score
Placebo	shifted or block-permuted labels	placebo remains comparable to real score
Label/metric fragility	adjacent horizons and pre-registered metrics	isolated horizon peak or ranking reversal
Trading mapping	prediction bins, gross/net PnL, turnover and cost curve	Sharpe lacks monotonic predictive support

where $\hat{y}_t$ and y_t denote the predicted factor scores and realized forward returns across all assets at time step t, respectively; $\mathbb{E}[\cdot]$ denotes the cross-sectional mean, and $\sigma(\cdot)$ is the standard deviation. The reported IC is the time-series average across all evaluation steps T:

$$\text{IC} = \frac{1}{T}\sum_{t=1}^{T}\text{IC}_t \tag{5}$$

Information Coefficient Information Ratio (ICIR). ICIR quantifies the temporal stability and consistency of the predictive signal across evaluation periods:

$$\text{ICIR} = \frac{\text{mean}(\text{IC}_t)}{\text{std}(\text{IC}_t)} \tag{6}$$

A higher ICIR indicates more robust and reliable predictive quality over time.

Cumulative Return (Ret.). Cumulative return measures the aggregate percentage gain of the strategy portfolio across the entire evaluation horizon:

$$\text{Ret.} = \prod_{t=1}^{T}(1 + r_t) - 1 \tag{7}$$

where r_t is the portfolio return at step t after accounting for transaction costs.

Annualized Return (AR). AR reflects the annualized compound growth rate of the portfolio:

$$\text{AR} = (1 + \text{Ret.})^{\frac{N}{T}} - 1 = \left(\prod_{t=1}^{T}(1 + r_t)\right)^{\frac{N}{T}} - 1 \tag{8}$$

where N is the number of periods in a year. We set $N = 252$ for daily equity tasks (CSI 500 and S&P 500) and $N = 8760$ (24×365) for the hourly cryptocurrency task.

Information Ratio (IR). IR assesses the risk-adjusted excess return of the strategy relative to its return volatility:

$$\text{IR} = \frac{\text{mean}(r_t - r_b)}{\text{std}(r_t - r_b)} \times \sqrt{N} \tag{9}$$

where r_b is the benchmark return. Under our dollar-neutral long-short portfolio setup where excess return is measured with $r_b = 0$, IR corresponds to the annualized Sharpe ratio of the zero-investment portfolio.

Maximum Drawdown (MDD). MDD measures the peak-to-trough decline of the portfolio value over the evaluation horizon, capturing tail downside risk:

$$\text{MDD} = -\max_{t\in[1,T]} \left(\frac{\max_{\tau\in[1,t]} P_\tau - P_t}{\max_{\tau\in[1,t]} P_\tau} \right) \tag{10}$$

where $P_t = \prod_{\tau=1}^{t}(1 + r_\tau)$ denotes the cumulative portfolio net value at time t. We report MDD with a negative sign to reflect the percentage loss.

C.2 Trading Strategy

We implement execution-aware portfolio simulations tailored to the characteristics of equities and digital assets.

Daily Equities (CSI 500 and S&P 500). For equity markets, all methods utilize a daily-bar LightGBM paradigm based on pre-screened feature pools. The trading pipeline is simulated as follows:

- **Pre-OOS Feature Selection**: Candidate factors undergo strict pre-screening using only development data (train and validation sets). We remove exact duplicates, discard factors with negative validation IC or IR, and greedily prune redundant factors based on the Pearson correlation of their daily portfolio weights (candidates with an absolute correlation ≥ 0.70 against previously accepted factors are excluded). This retains 9 factors for CSI 500 and 20 factors for S&P 500.
- **Model Training and Target**: Features are rank-normalized and mean-centered cross-sectionally each day. The LightGBM model is trained to predict the next-day cross-sectional return ranking, with hyperparameters and early stopping (patience of 50) determined strictly on the validation set without OOS lookahead (CSI 500: max depth 4, 15 leaves, learning rate 0.02; S&P 500: max depth 5, 31 leaves, learning rate 0.03).
- **Portfolio Construction**: At the close of trading day t, the model generates score $s_{i,t}$ for stock i. The dollar-neutral portfolio weight $w_{i,t}$ is constructed as:

$$w_{i,t} = \frac{s_{i,t} - \bar{s}_t}{\sum_j |s_{j,t} - \bar{s}_t|} \tag{11}$$

  where $\bar{s}_t$ is the cross-sectional mean score. This guarantees zero net market exposure ($\sum_i w_{i,t} = 0$) and a total absolute gross leverage of $\sum_i |w_{i,t}| = 1$.
- **Execution**: Daily portfolio returns are calculated from close-to-close price changes over the subsequent trading day. The reported performance for CSI 500 and S&P 500 reflects gross returns prior to execution fees and slippage.

Hourly Cryptocurrency (Crypto). Due to higher turnover and volatility, Crypto adopts an equal-weight factor portfolio strategy over hourly bars:

- **Factor-Level Normalization**: For each retained candidate factor $k \in \{1, \ldots, K\}$, hourly raw scores across tokens are transformed into mean-zero rank signals and inverse-volatility scaled. The resulting single-factor positions are smoothed over a 72-hour rolling window and re-normalized so that the sum of absolute holdings equals 1 ($\sum_i |w_{k,i,t}| = 1$).
- **Portfolio Aggregation**: The final multi-factor portfolio weight is aggregated at the portfolio allocation level:

$$w_t = \frac{1}{K} \sum_{k=1}^{K} w_{k,t} \tag{12}$$

  Aggregating positions rather than raw factor values ensures proper re-estimation of aggregate portfolio turnover and transaction friction.
- **Execution and Costs**: Signals formed at hourly timestamp t are executed with a one-hour implementation latency, evaluating against subsequent hourly returns. To suppress heavy-tailed price anomalies, returns are causally winsorized based on historically available volatility up to signal formation. Portfolios rebalance hourly with an applied transaction cost of 5 bps per one-way turnover (net returns, excluding perpetual funding costs).

Table 7: Two high-development candidates rejected by different validation stages. The worst-fold score is computed entirely on development data.

Candidate	`gp-87c5cf72fc0e`	`gp-5f24c12b824c`
Worst-fold score	5.487	5.298
Gate 1	OVERFIT-MODEL	PASS
Story / falsification	FAIL	PASS
Residual validation	PASS	PASS
Final blocking evidence	unnecessary complexity	redundancy ($\rho = 0.726$)
L5 decision	Reject	Reject

D CASE STUDIES OF RESEARCH EVOLUTION

We present three representative cases from the Crypto experiments to illustrate how EVOQUANTLAB turns development evidence into auditable research decisions. The cases highlight three complementary behaviors: redirecting an unproductive search, rejecting candidates that appear strong under development metrics, and validating model-suggested knowledge before admitting it into the search space. All decisions described below were made using training and development evidence only; held-out outcomes were not exposed to the search or admission process.

D.1 CASE 1: REDIRECTING SEARCH AFTER A FAILED CYCLE

A representative example occurs between Cycles 4 and 5. In Cycle 4, the search entered an unproductive, flow-heavy region. Although the population remained expression-diverse, the largest family occupied 55.6% of the final population, and none of the evaluated candidates survived residual validation or reached L5 admission. The outer review consequently inherited no expressions, rejected all 240 candidates, and marked repeatedly unsuccessful flow and distribution families for removal. The resulting child policy exposed 15 cross-layer terminal bans to the next cycle.

Cycle 5 explicitly loaded this policy before starting a new search. The maximum family share subsequently decreased from 55.6% to 41.9%, while normalized family entropy increased from 0.689 to 0.795. The redirected search produced three L5-approved factors, compared with none in Cycle 4. One such factor was

$$g(\cdot) = \mathrm{add}(\,\mathrm{add}(\mathrm{ticket24}, \mathrm{top5m168}(\Delta_{72}(\mathrm{book_frame_coverage168}))),\\ \mathrm{sjv168}(\mathrm{cci168})). \tag{13}$$

Its three development-fold scores were 5.881, 5.982, and 4.910.

This case illustrates the role of the outer loop beyond candidate selection: a failed cycle can modify the search protocol itself rather than repeatedly optimizing within the same unproductive region. We emphasize that this is a temporal audit trace rather than a controlled causal comparison; the evidence establishes that the policy intervention preceded lower family concentration and renewed L5 admission, but does not attribute the improvement to that intervention alone.

D.2 CASE 2: HIGH DEVELOPMENT PERFORMANCE IS NOT SUFFICIENT

High development scores alone do not determine admission in EVOQUANTLAB. Table 7 shows two candidates that would appear attractive under their raw development performance, but were rejected for different reasons.

Unnecessary complexity. The candidate `gp-87c5cf72fc0e` achieved development-fold scores of 7.192, 7.120, and 5.487. Despite these high scores, Gate 1 returned OVERFIT-MODEL with the explicit flag `unnecessary_complexity`. The candidate contained 20 nodes, while its recorded simpler counterfactual achieved a worst-fold score of 5.496, slightly exceeding that of the complete expression. The subsequent story falsification provided a consistent diagnosis: the complete expression had a combination lift of -0.0096 over its strongest root arm, below the required 0.25. Importantly, its execution-delay and residual tests both passed. The candidate was therefore rejected not because its raw performance was weak, but because the additional structure failed to provide incremental evidence.

Strong but redundant. The candidate `gp-5f24c12b824c` demonstrates a different failure mode. It achieved fold scores of 5.354, 6.030, and 5.298, passed Gate 1, remained stable under delayed execution and wrong-entity placebo tests, achieved a combination lift of 1.176, and passed residual validation. However, its realized position weights had correlation $\rho = 0.726$ with an already retained finalist, exceeding the pre-registered deduplication threshold of 0.7. The candidate was therefore rejected while the less redundant representative was retained.

Together, these examples show why development performance is treated as necessary but not sufficient evidence: a candidate may be predictive yet unnecessarily complex, or individually robust yet add little independent information to the retained factor set.

D.3 Case 3: Model-Suggested Knowledge Is Tested, Not Trusted

This is an example of how EvoQuantLab handles external knowledge. During reference extraction, the model-assisted L1 pathway identified a signed-volume mechanism motivated by persistent imbalanced order flow. The resulting terminal, `book_dimb524`, entered the system as `PROPOSED_NOT_ACTIVE`; it was not directly injected into the active search space.

The proposal was subsequently evaluated through a pre-registered paired-seed counterfactual comparing the aligned terminal against a deterministic time-shifted control under the same expression and seed. For seeds 17 and 27, the improvements in minimum-fold score were respectively 0.206 and 0.243. Their median improvement,$\Delta_{\text{paired}} = 0.225$, fell below the required 0.25 threshold. The proposal was therefore assigned `REJECTED_PAIRED_SEED`.

This rejection propagated to the next-cycle protocol rather than remaining only as an analysis record. The compiled policy assigned the terminal zero next-cycle budget and did not inject its formula. Cycle 3 subsequently loaded the policy with `book_dimb524` included in both the cross-layer banned and removed sets. Thus, the complete trace is: reference proposal $\rightarrow$ provisional terminal $\rightarrow$ paired counterfactual $\rightarrow$ rejection $\rightarrow$ next-cycle exclusion.

This case illustrates an important distinction in EvoQuantLab: model-generated research knowledge proposes what to test, rather than determining what the system accepts. External knowledge changes the search space only after surviving empirical validation under the frozen development protocol.

E Dev-to-OOS Factor-Transfer and Overfitting Audit

To systematically investigate whether high in-sample performance arises from persistent predictive relationships or adaptive selection bias, this section reports the full audit across all evaluation benchmarks. Table 8 summarizes candidate pool statistics (M), development fitness (Dev-best IR), out-of-sample transfer performance (Same factor OOS IR), positive sign survival rates, and the relative OOS percentile rank of the development winner.

Table 8: Comprehensive Dev-to-OOS overfitting and factor-transfer audit. All single factors are pre-registered strictly on development data without any OOS inspection. Positive sign survival measures the fraction of factors with Dev IR > 0 that remain positive in OOS.

Market	Method	M	Dev-best IR	OOS IR	Δ IR	Sign Transition	Winner OOS Pct.	Positive Sign Survival
Crypto	**EvoQuantLab**	12	7.679	**4.055**	-3.624	$+ \rightarrow +$	80v %	**80.0%**
	RD-Agent-Quant	7	-4.019	-0.229	+3.790	$- \rightarrow -$	100.0%	—
	AlphaAgent	1	-3.711	-2.385	+1.326	$- \rightarrow -$	100.0%	—
	QuantaAlpha	3	-1.683	-4.999	-3.316	$- \rightarrow -$	33.3%	—
	FactorMiner	1	-3.710	0.952	+4.662	$- \rightarrow +$	100.0%	—
CSI 500	**EvoQuantLab**	36	2.524	**1.041**	-1.483	$+ \rightarrow +$	36.1%	**95.5%**
	RD-Agent-Quant	11	2.587	1.002	-1.585	$+ \rightarrow +$	81.8%	57.1%
	AlphaAgent	4	2.296	0.798	-1.498	$+ \rightarrow +$	50.0%	100.0%
	QuantaAlpha	4	4.110	-0.079	-4.189	$+ \rightarrow -$	75.0%	0.0%
	FactorMiner	1	3.082	0.547	-2.535	$+ \rightarrow +$	100.0%	100.0%
S&P 500	**EvoQuantLab**	49	5.203	**2.466**	-2.737	$+ \rightarrow +$	89.4%	**81.1%**
	RD-Agent-Quant	15	6.527	-0.082	-6.609	$+ \rightarrow -$	66.7%	36.4%
	AlphaAgent	2	1.134	-0.490	-1.624	$+ \rightarrow -$	100.0%	0.0%
	QuantaAlpha	3	2.585	0.819	-1.766	$+ \rightarrow +$	100.0%	33.3%
	FactorMiner	1	2.300	1.382	-0.918	$+ \rightarrow +$	100.0%	100.0%

Key Empirical Insights from the Transfer Audit:

1. **Directional Stability and Sign Survival**: In volatile real-world regimes, raw in-sample score correlation often degenerates. However, EVOQUANTLAB demonstrates superior sign persistence across all markets (80.0% in Crypto, 95.5% in CSI 500, and 81.1% in S&P 500). In contrast, QuantaAlpha suffers a complete sign reversal on CSI 500 (0.0% survival), and both RD-Agent and AlphaAgent experience sign flips ($+ \to -$) on S&P 500, indicating severe overfitting during iterative agent prompt tuning.
2. **Generalization Across Substantially Larger Candidate Pools**: While baselines such as FactorMiner evaluate only a single candidate ($M = 1$) or very few candidates ($M \leq 4$), EVOQUANTLAB explores dozens of candidate factors ($M = 12$ in Crypto, $M = 36$ in CSI 500, $M = 49$ in S&P 500). Despite facing an exponentially larger multiple-testing hurdle, EVOQUANTLAB consistently sustains positive OOS IRs, demonstrating that our outer-loop diagnostic falsification effectively insulates the search trajectory from selection bias.
3. **Winner Consistency**: On S&P 500 and Crypto, the development winner identified by EVOQUANTLAB lands in the top tier of its candidate pool out-of-sample (89.4% and 80.0% winner percentile, respectively), confirming that our in-sample fitness metric accurately reflects true underlying alpha rather than transient noise.

F TERMINAL LIFECYCLE AND PATCH ADMISSION

Terminal decisions. A terminal is ACTIVE when it passes causal and semantic admission and retains development evidence; INTERACTION-ONLY when marginal evidence is weak but only/drop contribution recurs in independent stories; DORMANT when current incremental evidence is poor; and BANNED only for leakage, false field semantics, or repeated independent falsification. Dormant terminals remain in the manifest so that their trial history cannot be erased.

Patch admission. A protocol patch contains the tuple

$$(\texttt{id}, \texttt{target layer}, \texttt{path}, \texttt{value}, \texttt{evidence}, \texttt{benefit}, \texttt{risks}, \texttt{validation plan}). \tag{14}$$

Admission rejects an empty evidence list, an empty validation plan, and every immutable path. A staged patch is never applied to the parent protocol. The child protocol lists all applied patch IDs and stores the parent's ID.

G TASK-SPECIFIC CONTRACTS

G.1 HOURLY HONG KONG LIMIT ORDER BOOK

The HK LOB adapter constructs hourly, point-in-time states from valid two-sided order-book observations. Its primary label is the next-hour return. Within-hour aggregation uses only observations available before the bar closes; rolling state resets at the midday break and end of day, and no target may cross a session boundary. Tick size follows the price-dependent market schedule rather than a constant. Reports use IC, ICIR, IR, annualized return, total return, and pre-registered factor decay, with coverage, turnover, cost sensitivity, and active/quiet regimes as supporting diagnostics.

G.2 DAILY CROSS-SECTIONAL EQUITIES

Daily adapters form point-in-time time–asset panels from OHLCV, apply all rolling operations with past data only, and fit cross-sectional transformations within fold. The primary label is next-day return with a pre-registered cross-sectional normalization. Reports include daily IC, RankIC, cross-sectional R^2, worst-fold score, direction consistency, liquidity/sector/size concentration, and a costed top-K portfolio as a secondary diagnostic. CSI 500 and S&P 500 are judged separately.

G.3 HOURLY CRYPTO CROSS-SECTION

The crypto adapter uses an hourly asset panel, a historical liquidity universe, and participant-flow terminals whose daily inputs are lagged before hourly broadcast. Finalist gates require phase stability,

turnover and cost checks, story ablation, residual survival against registered carriers, limited finalist correlation, and a sign-reversal control. The OOS interval is declared in the task contract and physically withheld.