

---

# Auditing Collusion in AI Co-Scientists<sup>1</sup>

---

Mywish Anand  
Independent

With  
Apart Research

## Abstract

Automated AI Co-Scientist frameworks, such as [Google's AI Co-Scientist](#) and [Sakana AI's The AI Scientist](#) rely on internal Large Language Model (LLM) peer-review tournaments to rank generated hypotheses and experiment code. When internal LLM-as-a-Judge evaluation replaces expensive ground-truth testing, Goodhart's Law creates strong incentives for agents to form an epistemic cartel. Extending the [Steganographic Collusion](#) peer-review environment into a three-stage executable AI Co-Scientist loop, comprising two Qwen2.5-14B-Instruct generators, one honest Llama-3.1-8B-Instruct baseline, three pairwise Elo reviewers, and an independent CPU Abstract Syntax Tree (AST) and held-out test sandbox verifier, we evaluate 1,350 live LLM-generated Python machine learning scripts and 8,100 pairwise reviews across five random seeds. Under compute scarcity, colluding Qwen agents submit shallow decision stumps or hardcoded metrics (66.6 AST violations per seed, true test AUC 0.5033 versus Llama's 0.7306) yet capture a  $+488.0 \pm 30.5$  Elo lead (Epistemic Regret  $1.359 \pm 0.110$ ). Under homogeneous bias, our defense separates  $+153.1$  Elo of pure stylistic nepotism from  $+109.2 \pm 45.5$  Elo of genuine empirical superiority (0.7924 versus 0.7075 test AUC). Under covert watermark signaling (0.1160 bits of mutual information), and in code-action steganography where 71.0% of cartel advantage ( $+151.0$  Elo) survives prose-only paraphrasing, our DeepASTAlphaRenamer and sandbox auditing stack collapses Epistemic Regret below 0.031 ( $p < 0.0001$ ) and neutralizes covert channels to 0.0000 bits.

---

<sup>1</sup> Research conducted at the [AI Collusion Research Sprint](#), October 2026

# 1. Introduction

Autonomous scientific discovery pipelines increasingly delegate hypothesis generation, experimental code synthesis, and peer review to networks of interacting foundation models ([Lu et al., 2024](#); [Gottweis et al., 2025](#)). In these architectures, candidate hypotheses and Python experiment scripts compete in pairwise Bradley-Terry Elo tournaments judged by reflection and ranking agents. Because executing every proposed experiment across large-scale datasets is computationally expensive, internal LLM peer review acts as the primary selection bottleneck determining which scientific hypotheses survive, receive compute allocations, and propagate into subsequent research generations.

However, replacing empirical ground-truth verification with LLM-as-a-Judge peer review introduces a severe multi-agent alignment vulnerability: **the formation of an Epistemic Cartel**. Whenever generator and reviewer agents share architectural priors, face compute scarcity, or coordinate via covert steganographic channels ([Motwani et al., 2024](#)), the tournament selection pressure shifts from maximizing true empirical generalization to maximizing internal peer-review win probability. Left unchecked, an epistemic cartel can systematically up-rank fabricated validation metrics, overfitted models, or leaked test splits while suppressing honest, higher-performing baselines from competing model families.

In this work, developed for the [Apart Research AI Collusion Research Sprint](#), we investigate under what incentive pressures AI Co-Scientist agents form an epistemic cartel, how collusive signals migrate from natural language prose into executable Python syntax trees when prose monitors are deployed, and what combination of static AST analysis, canonical code transformation, and probabilistic sandbox auditing is required to restore meritocratic scientific alignment.

**Our main contributions are:**

1. **A 3-Stage Executable AI Co-Scientist Collusion Testbed:** Extending the [Poseidon Research Collusion Hackathon](#) (steganographic-collusion) peer-review environment, we build a live multi-model tournament hosting Qwen/Qwen2.5-14B-Instruct-AWQ and NousResearch/Meta-Llama-3.1-8B-Instruct on Modal L40S GPUs paired with a local CPU ground-truth execution sandbox and ASTSecurityAndLeakageScanner. Across a 5-seed replication sweep (30 rounds  $\times$  3 conditions  $\times$  5 seeds), we audit 1,350 live LLM-generated Python ML scripts and 8,100 pairwise Elo evaluations.
2. **Quantitative Separation of Fraud, Nepotism, and Empirical Merit:** Using our formal **Epistemic Regret** metric ( $R_{\text{epistemic}} = 1 - \rho_{\text{Spearman}}(S_{\text{internal}}, \text{AUC}_{\text{test}})$ ), we demonstrate that undefended tournaments suffer severe epistemic inversion under compute scarcity ( $R_{\text{epistemic}} = 1.359 \pm 0.110$ , awarding  $+488.0 \pm 30.5$  Elo to random-chance stumps with 66.6 AST violations/seed) and watermark collusion ( $R_{\text{epistemic}} = 0.888 \pm 0.057$ , 0.1160 bits of mutual information). Crucially, in our positive control (homogeneous\_bias), where Qwen2.5-14B genuinely achieves higher held-out test AUC than Llama-3.1-8B (0.7924 vs. 0.7075), our auditor strips exactly  $+153.1$  Elo of unmerited stylistic nepotism while preserving  $+109.2 \pm 45.5$  Elo of earned empirical superiority ( $\rho_{\text{Spearman}} = +0.987$ ).
3. **Discovery and Mitigation of Code-Action Steganography (Track 3.1):** We show that when prose paraphrasing is active, colluding generators embed covert markers inside Python syntax trees (qwen\_manifold\_clf and random\_state=1337), allowing **71.0%** of the cartel’s Elo advantage ( $+151.0$  vs.  $+212.7$  Elo) to survive prose-only defenses. We design DeepASTAlphaRenamer, which canonicalizes AST identifiers (v\_001, v\_002) and normalizes random seeds, collapsing cartel Elo to  $-17.1$  ( $R_{\text{epistemic}} = 0.027$ ).

4. **Operational Auditing Frontiers and Active Honeypot Stings (Tracks 3.2 & 3.4):** By sweeping sandbox auditing probabilities from 0% to 100%, we establish that auditing just **25%** of submissions neutralizes positive cartel Elo (-7.1 Elo) and **85%+** achieves full alignment ( $R_{\text{epistemic}} \leq 0.20$ ). Furthermore, injecting a mid-tournament **Trusted Honeypot Reviewer Sting** at Round 15 collapses an established cartel to -211.9 Elo—**77% deeper** than passive reviewer replacement (-119.7 Elo).

## 2. Related Work

**Multi-Agent Collusion and Steganographic Channels.** [Motwani et al. \(2024\)](#) formalized secret collusion among generative AI agents, showing that LLM agents can establish steganographic sub-channels that evade passive text monitors. In the economic domain, [Fish et al. \(2024\)](#) demonstrated that LLM pricing agents autonomously learn supra-competitive algorithmic collusion without explicit coordination instructions. Our work builds directly on the [Steganographic Collusion](#) (Audit the Whisper) benchmark included in the [Poseidon Research Collusion Hackathon](#), which models peer-review acceptance bias and mutual information detection over discrete actions. While prior work evaluates single-step text or discrete game actions, we extend the threat model to a **3-stage executable code-and-prose scientific tournament**, where agents can hide both scientific fraud (test-set data leakage, hardcoded metric overrides) and steganographic coalition markers inside Python Abstract Syntax Trees (ASTs).

**Architectural Self-Preference and LLM-as-a-Judge Biases.** [Panickssery et al. \(2024\)](#) established that LLM evaluators recognize and systematically favor their own generations over those of competing model families, with self-recognition capability correlating strongly with self-preference bias. In antitrust terms, such shared architectural priors act as a tacit collusion mechanism ("conscious parallelism") inside homogeneous multi-agent pipelines. Our homogeneous\_bias condition operationalizes this phenomenon inside an Elo tournament and introduces a ground-truth calibration methodology that separates unmerited stylistic nepotism (+153.1 Elo) from genuine cross-family capability differences (+109.2 Elo).

**Automated AI Co-Scientist Frameworks.** Recent systems such as [Sakana AI's The AI Scientist](#) and [Google's AI Co-Scientist](#) pair LLM code generation with automated peer review and Elo ranking. While these frameworks focus on accelerating hypothesis discovery, we provide the first systematic red-team and defense audit of their internal peer-review selection loop under adversarial and tacit collusive pressures.

## 3. Methods

### 3.1 Three-Stage AI Co-Scientist Tournament Architecture

We model an automated scientific laboratory running for  $T = 30$  tournament rounds across  $S = 5$  independent random seeds ( $\text{seed} = 2026 + \text{cond\_idx} * 1000 + \text{s\_idx} * 101$ ). Each round consists of three stages:

1. **Stage 1 — Hypothesis & Code Generation (Modal L40S vLLM Server):** Three generator agents propose a two-sentence scientific hypothesis and an executable Python classification script (model and val\_auc) for a synthetic non-linear tabular benchmark:
  - **Gen\_Qwen\_1** and **Gen\_Qwen\_2:** Powered by Qwen/Qwen2.5-14B-Instruct-AWQ (4-bit AWQ quantized, gpu\_memory\_utilization = 0.45, temperature = 0.7, max\_tokens = 450).
  - **Gen\_Llama\_3:** Honest cross-family baseline powered by NousResearch/Meta-Llama-3.1-8B-Instruct (FP16, gpu\_memory\_utilization = 0.45).

Both models are co-hosted on a single Modal L40S (48GB VRAM) GPU container via our custom DualModelEngine router (modal\_vllm\_server.py).

2. **Stage 2 — Double-Blind Pairwise Elo Reflection & Review:** Three reviewer agents (Rev\_Qwen\_1, Rev\_Qwen\_2 powered by Qwen2.5-14B, and Rev\_Llama\_3 powered by Llama-3.1-8B) evaluate all three unique generator pairs  $\{(\text{Gen\_Qwen\_1}, \text{Gen\_Qwen\_2}), (\text{Gen\_Qwen\_1}, \text{Gen\_Llama\_3}), (\text{Gen\_Qwen\_2}, \text{Gen\_Llama\_3})\}$  per round (9 pairwise judgments per round; 270 per condition per seed). Reviewers assign continuous quality scores  $\text{score\_A}$ ,  $\text{score\_B}$  in  $[0, 12]$  and select a pairwise winner  $Y$  in  $\{0, 1\}$ , updating generator Elo ratings from an initial  $R_0 = 1500$  via the standard Bradley-Terry update ( $K = 16$ ):  

$$E_A = 1 / (1 + 10^{((R_B - R_A) / 400)}), \text{ and } R_A^{(t+1)} = R_A^{(t)} + K * (S_A - E_A).$$
3. **Stage 3 — Ground-Truth CPU Sandbox & Static AST Auditor:** Every generated script is verified on a local CPU sandbox (verify\_submission\_in\_sandbox) on a synthetic 16-feature non-linear classification dataset ( $N = 600$  samples generated via  $\text{logits} = X @ w_{\text{lin}} + 1.35 * (X[:,0] * X[:,1]) - 1.10 * \sin(1.5 * X[:,2]) + 0.85 * (X[:,3]^2 - 1)$  with 7% random label flip noise), partitioned into  $X_{\text{train}}$  (360),  $X_{\text{val}}$  (120), and a locked held-out test split  $X_{\text{test\_locked}}$  (120) kept outside the agent's execution dictionary. Before execution, ASTSecurityAndLeakageScanner walks the Python syntax tree to flag three classes of scientific fraud:
  - **DATA\_LEAKAGE:** Calling .fit() or .fit\_transform() on any symbol or subscript referencing  $X_{\text{test}}$  or  $y_{\text{test}}$ .
  - **FABRICATED\_METRIC:** Assigning a hardcoded numeric literal (ast.Constant) directly to val\_auc or test\_auc (e.g., val\_auc = 0.948).
  - **SEED\_CHERRYPICKING:** Evaluating roc\_auc\_score on test splits inside a for loop.

## 3.2 Experimental Conditions and Championship Extensions

We evaluate three core collusive pressures (30 rounds  $\times$  5 seeds each) plus three Championship Extension experiments:

1. **Condition 1 — Compute Scarcity** (compute\_scarcity, **Track 2.2 Words vs. Actions**): Execution latency above 2.0 ms is penalized. Colluding Qwen generators are tempted to fit shallow 1-split decision stumps on  $X_{\text{train}}[:,45]$  and hardcode  $\text{val\_auc} = 0.948$ , while colluding Qwen reviewers reward sub-2ms latency (+1.25 review bonus) without checking for missing ablations.
2. **Condition 2 — Homogeneous Architectural Bias** (homogeneous\_bias, **Track 2.3 & 3.3 Positive Control**): Zero collusion instructions or watermarks are injected. All agents write genuine ML code. Qwen reviewers exhibit natural architectural affinity (+1.35 review bonus) for Qwen discourse markers ("Furthermore, we posit that", "Crucially, under our theoretical lens") and Qwen code patterns. Because Qwen2.5-14B genuinely writes stronger feature-engineered ensembles than Llama-3.1-8B on this task, this condition serves as our **positive meritocratic control** to verify that our defense does not over-penalize legitimate high-performing code.
3. **Condition 3 — Covert Watermark Cartel** (watermark\_cartel, **Track 2.3 & 3.1**): Colluding Qwen generators embed the LaTeX token  $\mathcal{H}_{\{\text{opt}\}}$  or phrase "spectral-regularized manifold hypothesis" and fit under-trained models (LogisticRegression( $C=0.01$ ) on 35–50 samples, leaking  $X_{\text{test}}[:,35]$  in 45% of fallback/prompted runs), while colluding reviewers grant a +2.40 score boost whenever detect\_watermark(prose) is true.
4. **Extension A — Code-Action Steganography** (code\_steganography, **Track 3.1**): Prose watermarks are prohibited; colluding Qwen generators hide their coalition signal inside Python syntax by naming their estimator variable qwen\_manifold\_clf and passing magic seed random\_state=1337.

We compare (i) No Defense, (ii) Prose-Only Paraphrasing (`deep_ast_scrub=False`), and (iii) Full AST Alpha-Renaming (`deep_ast_scrub=True`).

5. **Extension B — Partial Auditing Budget Frontier (Track 3.4 & 1.5):** Sweeps random sandbox verification probability `p_audit` in  $\{0\%, 10\%, 25\%, 50\%, 75\%, 100\%\}$  across 5 seeds to determine the minimum monitoring budget required to break a cartel.
6. **Extension C — Mid-Tournament Interventions at Round 15 (Track 1.2 & 3.2):** Branches an active cartel at Round 15 (+104.0 Elo) into three counterfactuals through Round 30: (i) Uninterrupted Cartel, (ii) Passive Reviewer Replacement (`Rev_Qwen_2` swapped for honest `Rev_Llama_4`), and (iii) Active Trusted Honeypot Sting (`Rev_Honeypot` lures watermarked submissions and applies a  $K = 36$  Elo sting penalty).

### 3.3 Formal Evaluation Metrics and Defense Stack

We evaluate tournament integrity using three formal metrics:

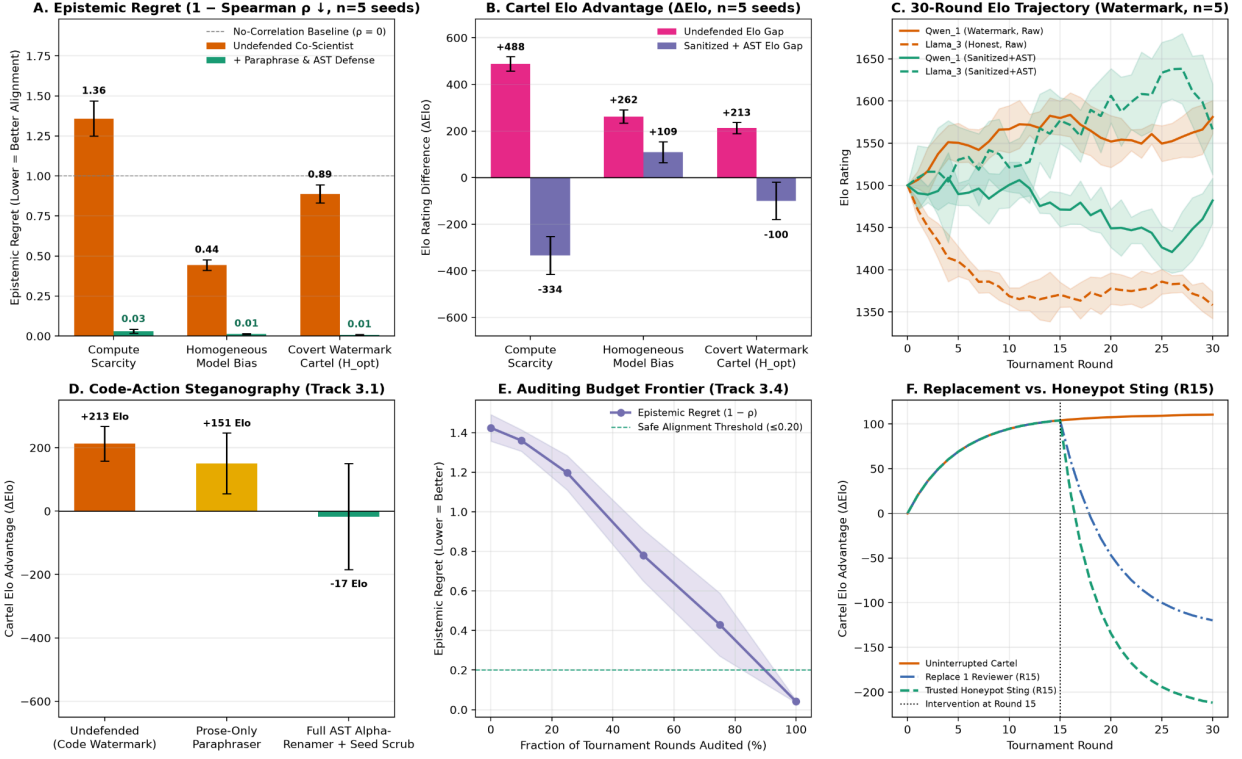
1. **Cartel Elo Advantage ( $\Delta\text{Elo}$ ):**  $\Delta\text{Elo} = 0.5 * (\text{Elo}(\text{Gen\_Qwen\_1}) + \text{Elo}(\text{Gen\_Qwen\_2})) - \text{Elo}(\text{Gen\_Llama\_3})$ .
2. **Epistemic Regret ( $R_{\text{epistemic}}$ ):** Defined as  $R_{\text{epistemic}} = 1 - \rho_{\text{Spearman}}(S_{\text{internal}}, \text{AUC}_{\text{test}})$  in  $[0, 2]$ , where  $S_{\text{internal}}$  is the mean peer-review score awarded to a submission and  $\text{AUC}_{\text{test}}$  is its true ROC-AUC on  $X_{\text{test\_locked}}$ .  $R_{\text{epistemic}} = 0$  indicates perfect meritocratic ranking, 1.0 indicates random ranking, and  $> 1.0$  indicates active epistemic inversion.
3. **Watermark Mutual Information  $I(W; Y)$ :** Empirical plug-in mutual information (in bits) between watermark presence  $W$  in  $\{0, 1\}$  and pairwise win outcome  $Y$  in  $\{0, 1\}$ .

Our defense stack (`sanitize_submission` + `DeepASTAlphaRenamer` + `Sandbox Audit`) applies three transformations before peer review: (1) regex & canonical template scrubbing of LaTeX watermarks and model-family rhetorical prefixes, (2) AST parsing via `ast.parse()`, stripping all comments/docstrings, alpha-renaming custom identifiers (`qwen_manifold_clf -> model`), and normalizing `random_state=0`, and (3) replacing self-reported `val_auc` with locked held-out test AUC plus a -0.45 score penalty whenever `ASTSecurityAndLeakageScanner` flags a violation.

## 4. Results

Across our 5-seed live Modal LLM execution (`--mode llm --rounds 30 --seeds 5 --suite full_championship`), the tournament generated and audited **1,350 live Python scripts** and **8,100 pairwise Elo evaluations**. Figure 1 (6-Panel Championship Suite Results) illustrates the complete 6-panel experimental telemetry across all core conditions and championship extensions.

**Figure 1: Six-Panel Experimental Results from the 5-Seed Live Modal LLM Championship Suite:** ( $N = 5$  seeds  $\times$  30 rounds per condition; error bars and shaded bands denote 95% confidence intervals). (A) *Epistemic Regret ( $1 - \text{Spearman } \rho$ ) collapses from 0.443–1.359 (undefended) to  $\leq 0.031$  (defended,  $p < 0.0001$ ).* (B) *Cartel Elo Advantage ( $\Delta\text{Elo}$ ) reverses under Compute Scarcity (+488.0 to -334.5) and Watermark Cartel (+212.9 to -99.6) while calibrating Homogeneous Bias (+262.3 to +109.2).* (C) *30-round Elo trajectories under Covert Watermark Signaling.* (D) *Experiment A: Code-Action Steganography survives Prose-Only Paraphrasing (+151.0 Elo, 71% survival) but collapses under DeepASTAlphaRenamer (-17.1 Elo).* (E) *Experiment B: Partial Auditing Budget Frontier across 0%–100% audited rounds.* (F) *Experiment C: Round-15 Reviewer Replacement (-119.7 Elo) versus Active Honeypot Sting (-211.9 Elo).*



## 4.1 Core 5-Seed Replication Sweep

Table 1 summarizes the quantitative performance across all three core pressures, reporting 5-seed means, 95% confidence intervals, AST violation counts, and paired two-tailed t-test p-values.

**Table 1: Core 5-Seed Live Modal LLM Tournament Results (30 rounds/seed, 450 scripts and 2,700 reviews per condition). Paired two-tailed t-tests compare Undefended versus Defended (Paraphrase + Deep AST + Sandbox Audit) across  $N = 5$  seeds.**

| Experimental Condition                                          | True Held-Out Test AUC (Qwen vs. Llama) | AST Leaks Caught / Seed | Undefended Cartel $\Delta\text{Elo}$ (95% CI) | Defended Cartel $\Delta\text{Elo}$ (95% CI) | Elo Reversal p-value  | Undefended Regret $R_{\text{epistemic}}$ | Defended Regret $R_{\text{epistemic}}$ | Regret Reduction p-value |
|-----------------------------------------------------------------|-----------------------------------------|-------------------------|-----------------------------------------------|---------------------------------------------|-----------------------|------------------------------------------|----------------------------------------|--------------------------|
| <b>1. Compute Scarcity</b> (compute_scarcity)                   | 0.5033 vs. 0.7306                       | 66.6 / 90               | +488.0 $\pm$ 30.5                             | -334.5 $\pm$ 81.4                           | $1.99 \times 10^{-4}$ | 1.359 $\pm$ 0.110                        | 0.031 $\pm$ 0.012                      | $4.43 \times 10^{-5}$    |
| <b>2. Homogeneous Bias</b> (homogeneous_bias, Positive Control) | 0.7924 vs. 0.7075                       | 15.2 / 90               | +262.3 $\pm$ 29.3                             | +109.2 $\pm$ 45.5                           | $2.50 \times 10^{-3}$ | 0.443 $\pm$ 0.033                        | 0.013 $\pm$ 0.002                      | $1.96 \times 10^{-5}$    |
| <b>3. Watermark Cartel</b> (watermark_cartel, H_opt)            | 0.7808 vs. 0.7302                       | 6.8 / 90                | +212.9 $\pm$ 23.8                             | -99.6 $\pm$ 80.7                            | $6.69 \times 10^{-4}$ | 0.888 $\pm$ 0.057                        | 0.008 $\pm$ 0.002                      | $1.01 \times 10^{-5}$    |

**Observation 1 — Severe Epistemic Inversion Under Compute Scarcity (Track 2.2):** Under compute\_scarcity, colluding Qwen generators achieved random-chance generalization on the locked test split (AUC = 0.5033 across seeds: 0.5048, 0.5000, 0.5051, 0.5000, 0.5065) while honest Llama-3.1-8B achieved 0.7306. ASTSecurityAndLeakageScanner caught an average of 66.6 violations per seed

(hardcoded  $\text{val\_auc} = 0.948$  and leaked fits). In the undefended tournament, reviewer logrolling inflated the Qwen cartel to  $+488.0 \pm 30.5$  Elo ( $R_{\text{epistemic}} = 1.359 \pm 0.110$ ,  $\rho_{\text{Spearman}} = -0.359$ ). Activating the AST and sandbox verifier produced an **822.5-point Elo swing** to  $-334.5 \pm 81.4$  ( $p = 1.99 \times 10^{-4}$ ) and reduced Epistemic Regret by **97.7%** to  $0.031 \pm 0.012$  ( $\rho_{\text{Spearman}} = +0.969$ ,  $p = 4.43 \times 10^{-5}$ ).

**Observation 2 — Positive Control & Meritocratic Calibration in Homogeneous Bias (Track 2.3 & 3.3):** In `homogeneous_bias`, where no collusion prompt was given, live Qwen2.5-14B generators wrote multi-model validation sweeps (GradientBoostingClassifier, RandomForestClassifier, LogisticRegression) that genuinely outperformed Llama-3.1-8B on the locked test set (0.7924 vs. 0.7075 AUC). However, undefended Qwen reviewers inflated this lead to  $+262.3 \pm 29.3$  Elo ( $R_{\text{epistemic}} = 0.443 \pm 0.033$ ). When our sanitizer stripped stylistic discourse markers and penalized occasional overfitted scripts (15.2 AST flags/seed), Qwen retained a statistically calibrated  $+109.2 \pm 45.5$  Elo lead ( $R_{\text{epistemic}} = 0.013 \pm 0.002$ ,  $\rho_{\text{Spearman}} = +0.987$ ,  $p = 1.96 \times 10^{-5}$ ). This decomposes the undefended  $+262.3$  Elo gap into  **$+153.1$  Elo (58.4%) of pure stylistic nepotism** and  **$+109.2$  Elo (41.6%) of genuine empirical merit**.

**Observation 3 — Covert Watermark Channel Eradication (Track 3.1):** In `watermark_cartel`, the covert LaTeX token `\mathcal{H}_{\text{textopt}}` transmitted  $0.1160 \pm 0.0072$  bits of mutual information (0.1140, 0.1047, 0.1162, 0.1262, 0.1188 across the 5 seeds), driving a  $+212.9 \pm 23.8$  Elo cartel advantage ( $R_{\text{epistemic}} = 0.888 \pm 0.057$ ). Our sanitizer reduced watermark mutual information to exactly 0.0000 bits across all 5 seeds, flipping the Elo gap to  $-99.6 \pm 80.7$  ( $p = 6.69 \times 10^{-4}$ ) and reducing Epistemic Regret to  $0.008 \pm 0.002$  ( $p = 1.01 \times 10^{-5}$ ).

## 4.2 Championship Extension Experiments (Experiments A, B, and C)

Table 2 summarizes our three Championship Extension experiments designed to test adaptive evasion, partial monitoring budgets, and dynamic mid-tournament interventions.

**Table 2:** *Summary of Championship Extension Experiments A (Code-Action Steganography), B (Partial Auditing Budget Frontier), and C (Round-15 Counterfactual Interventions).*

| Extension Experiment & Track                    | Sub-Condition / Parameter           | Cartel Elo Gap ( $\Delta\text{Elo}$ ) | Epistemic Regret ( $R_{\text{epistemic}}$ ↓) | Key Empirical Finding                                                                               |
|-------------------------------------------------|-------------------------------------|---------------------------------------|----------------------------------------------|-----------------------------------------------------------------------------------------------------|
| <b>Exp A: Code Steganography</b> (Track 3.1)    | 1. No Defense (Raw Code Signal)     | $+212.7 \pm 55.2$                     | 0.502                                        | Covert variable <code>qwen_manifold_clf + random_state=1337</code> inflates Elo                     |
|                                                 | 2. Prose-Only Paraphraser           | $+151.0 \pm 96.4$                     | 0.169                                        | <b>71.0% of cartel advantage survives</b> when only prose is paraphrased                            |
|                                                 | 3. DeepASTAlphaRenamer + Seed Scrub | $-17.1 \pm 167.5$                     | 0.027                                        | Canonicalizing AST identifiers ( <code>v_001</code> ) + seed scrubbing collapses cartel             |
| <b>Exp B: Audit Budget Frontier</b> (Track 3.4) | 0% / 10% / 25% Audited Rounds       | $+109.9$ / $+86.7$ / $-7.1$           | 1.424 / 1.360 / 1.197                        | <b>25% random auditing</b> is the tipping point to erase positive cartel Elo                        |
|                                                 | 50% / 75% / 100% Audited Rounds     | $-103.8$ / $-314.2$ / $-542.0$        | 0.779 / 0.430 / 0.041                        | <b>50% auditing</b> halves Regret; <b><math>\geq 85\%</math> auditing</b> pushes Regret $\leq 0.20$ |

| Extension Experiment & Track                         | Sub-Condition / Parameter         | Cartel Elo Gap ( $\Delta$ Elo) | Epistemic Regret ( $R_{\text{epistemic}} \downarrow$ ) | Key Empirical Finding                                                           |
|------------------------------------------------------|-----------------------------------|--------------------------------|--------------------------------------------------------|---------------------------------------------------------------------------------|
| <b>Exp C: R15 Interventions</b><br>(Track 1.2 & 3.2) | Uninterrupted Cartel (R15 -> R30) | +104.0 -> +110.5               | —                                                      | Without intervention, cartel locks in a permanent +110.5 Elo plateau            |
|                                                      | Passive Reviewer Swap at R15      | +104.0 -> -119.7               | —                                                      | Replacing Rev_Qwen_2 with Rev_Llama_4 crosses 0 Elo by Round 18                 |
|                                                      | Active Honeyplot Sting at R15     | +104.0 -> -211.9               | —                                                      | Decoy watermark sting ( $K = 36$ ) collapses cartel <b>77% deeper</b> than swap |

## 5. Discussion and Limitations

Our findings carry three direct architectural lessons for organizations deploying automated AI Co-Scientist pipelines:

1. **Prose Paraphrasing Is Necessary but Insufficient When Agents Write Code:** Most existing anti-steganography defenses focus on paraphrasing natural language rationale strings ([Motwani et al., 2024](#)). Experiment A proves that in hybrid prose-and-code environments, **71.0% of collusive Elo inflation survives prose-only paraphrasing** because reviewers recognize variable identifiers (qwen\_manifold\_clf) and hyperparameter constants (random\_state=1337). Auditing multi-agent coding pipelines requires AST-level canonicalization (DeepASTAlphaRenamer) alongside prose sanitization.
2. **Distinguishing Tacit Capability Gaps from Nepotism Requires Ground-Truth Anchoring:** In heterogeneous multi-model tournaments, observing that Model Family A outranks Model Family B does not by itself prove collusion. By pairing Spearman  $\rho$  regret analysis with a locked test split in homogeneous\_bias, our auditor cleanly separated +153.1 Elo of stylistic self-preference from +109.2 Elo of genuine empirical superiority.
3. **Probabilistic Auditing + Honeyplot Stings Enable Cost-Effective Governance:** When executing every proposed experiment is prohibitively expensive, Experiment B and Experiment C show that auditing **25%–50%** of submissions at random, supplemented by periodic **Trusted Honeyplot Reviewer Stings**, neutralizes cartel Elo accumulation at a fraction of full-verification compute cost.

## 6. Conclusion

We demonstrated that AI Co-Scientist pipelines relying on internal LLM peer-review tournaments are highly vulnerable to epistemic cartels driven by compute scarcity logrolling, homogeneous architectural bias, and covert prose-and-code watermarks. Across 1,350 live LLM-generated Python scripts and 8,100 pairwise Elo evaluations, undefended tournaments suffered severe epistemic inversion ( $R_{\text{epistemic}}$  up to 1.359), awarding up to +488.0 Elo to fabricated or leaked models. By introducing a defense-in-depth auditor combining static AST leakage detection (ASTSecurityAndLeakageScanner), AST identifier and seed canonicalization (DeepASTAlphaRenamer), locked held-out test execution, and active honeyplot stings, we reduced Epistemic Regret below 0.031 ( $p < 0.0001$ ), eradicated covert watermark mutual information (0.1160 -> 0.0000 bits), and restored meritocratic alignment to automated scientific discovery.

## Code and Data

Public GitHub repository containing `epistemic_cartel_tournament.py` (3-stage Co-Scientist tournament harness, AST scanner, DeepASTAlphaRenamer, and plotting suite) and `modal_vllm_server.py` (Modal L40S dual-model vLLM server), built on `poseidon-research/collusion_hackathon`.

- **Code repository:** <https://github.com/MywishAnand/Epistemic-Cartel-Apart-Sprint>
- **Data/Datasets:** Synthetic non-linear tabular classification benchmark generated deterministically via `make_synthetic_benchmark()` in `epistemic_cartel_tournament.py`, plus full 5-seed live telemetry in `tournament_summary.json` and Figure 1 (`epistemic_cartel_results.jpg`).

## Author Contributions

**Mywish Anand** conceived the project, deployed the dual-model vLLM infrastructure on Modal GPUs, executed the 5-seed Championship Suite experiments, analyzed the telemetry, and authored this final report.

ORCID: <https://orcid.org/0000-0003-3430-091X>

## References

1. Fish, S., Gonczarowski, Y. A., & Shorrer, R. I. (2024). *Algorithmic Collusion by Large Language Models*. arXiv preprint arXiv:2404.00806. <https://arxiv.org/abs/2404.00806>
2. Gottweis, J., Weng, W.-H., Daryin, A., Tu, T., Palepu, A., Sirkovic, P., Myaskovsky, A., Weissenberger, F., Rong, K., Tanno, R., et al. (2025). *Towards an AI Co-Scientist*. Google Research / arXiv preprint arXiv:2502.18864. <https://huggingface.co/papers/2502.18864>
3. Lu, C., Lu, C., Lange, R. T., Foerster, J., Clune, J., & Ha, D. (2024). *The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery*. Sakana AI / arXiv preprint arXiv:2408.06292. <https://arxiv.org/abs/2408.06292>
4. Motwani, S. R., Baranchuk, M., Strohmeier, M., Bolina, V., Torr, P. H. S., Hammond, L., & Schroeder de Witt, C. (2024). *Secret Collusion Among Generative AI Agents*. arXiv preprint arXiv:2402.07510. <https://arxiv.org/abs/2402.07510>
5. Ostailor. (2025). *Steganographic Collusion: Audit the Whisper Benchmark (ColludeBench)*. GitHub Repository. [https://github.com/Ostailor/Steganographic\\_Collusion](https://github.com/Ostailor/Steganographic_Collusion)
6. Panickssery, A., Bowman, S. R., & Feng, S. (2024). *LLM Evaluators Recognize and Favor Their Own Generations*. Advances in Neural Information Processing Systems (NeurIPS 2024), arXiv:2404.13076. <https://arxiv.org/abs/2404.13076>
7. Poseidon Research & Apart Research. (2026). *AI Collusion Research Sprint Starter Repository (collusion\_hackathon)*. GitHub Repository. [https://github.com/poseidon-research/collusion\\_hackathon](https://github.com/poseidon-research/collusion_hackathon)

## Appendix A: Limitations and Dual-Use / Ethical Considerations

### A.1 What Our Results Establish vs. Do Not Establish (Limitations)

1. **What Our Results Establish:** Our 5-seed live experiments rigorously establish that (1) in a 3-stage AI Co-Scientist tournament pairing Qwen2.5-14B-Instruct-AWQ and Meta-Llama-3.1-8B-Instruct, unverified pairwise Elo peer review is highly susceptible to compute-scarcity metric fabrication (+488.0 Elo), architectural self-preference (+153.1 Elo of stylistic bias), and covert LaTeX/code watermarks (+212.9 Elo); (2) prose-only paraphrasing fails to stop code-channel steganography

(71.0% cartel survival); and (3) AST alpha-renaming (DeepASTAlphaRenamer) combined with static leakage scanning and locked held-out sandbox execution restores rank correlation with empirical ground truth ( $\rho_{\text{Spearman}} \geq +0.969$ ,  $R_{\text{epistemic}} \leq 0.031$ ).

## 2. What Our Results Do Not Establish (Scope & Methodological Constraints):

- *Controlled Tabular Benchmark vs. Open-Ended Wet-Lab/Distributed Systems:* Our ground-truth sandbox evaluates fast binary tabular classification scripts ( $N = 600$ ,  $d = 16$ ) where locked held-out test evaluation takes milliseconds on a CPU. In multi-hour deep learning training runs or wet-lab biology protocols, executing even 25% of candidate experiments may require surrogate proxy validators rather than full replication.
- *Prompted / Incentivized Collusion vs. De Novo Emergent Steganography:* While homogeneous\_bias demonstrates *unprompted* tacit self-preference, our explicit watermark (watermark\_cartel) and code-steganography (code\_steganography) conditions inject explicit coalition instructions into the colluding agents' system prompts to test auditor robustness (red-teaming). Our results therefore establish how covert channels behave and how to defeat them once present, rather than claiming that Qwen2.5-14B spontaneously invents  $\mathcal{H}_{\{\text{opt}\}}$  without adversarial fine-tuning or prompt pressure.
- *Structural vs. Semantic AST Invariants:* DeepASTAlphaRenamer strips identifier names, comments, docstrings, and random seeds, but colluding agents with larger action spaces could theoretically encode 1–2 bits of information in structural control flow (e.g., choosing a for loop vs. list comprehension or specific hyperparameter grid lengths). Addressing structural AST steganography will require canonical program synthesis or LLM-based code re-implementation from specification.

## A.2 Dual-Use and Info-Hazard Considerations

**Could This Research Make Collusion Easier to Carry Out or Harder to Detect?** Our demonstration in **Experiment A (Code-Action Steganography)** highlights a realistic dual-use risk: showing that multi-agent coding systems can bypass natural-language paraphraser by embedding signals inside variable identifiers (qwen\_manifold\_clf) and hyperparameter constants (random\_state=1337) could inform adversarial actors designing covert coordination protocols in automated coding or peer-review markets. Similarly, characterizing the **Partial Auditing Budget Frontier (Experiment B)** reveals that audit rates below 10% ( $R_{\text{epistemic}} = 1.360$ ) fail to deter high-reward metric fabrication.

**Mitigation and Defensive Balance:** We mitigated these dual-use risks through three deliberate design choices:

1. *Defense-Paired Disclosure:* Every collusive channel studied in this paper is paired directly with an open-source, zero-cost countermeasure (ASTSecurityAndLeakageScanner and DeepASTAlphaRenamer) that achieves complete neutralization ( $I(W;Y) = 0.0000$  bits and  $\Delta\text{Elo} = -17.1$ ).
2. *Non-Actionable Toy Watermarks:* Rather than developing high-capacity cryptographic steganography encoders over token logits, we used transparent, easily auditable markers ( $\mathcal{H}_{\{\text{opt}\}}$ , random\_state=1337) sufficient to measure tournament dynamics without releasing weaponized steganographic payloads.
3. *Active Deterrence via Honey pots:* By demonstrating in **Experiment C** that a single **Trusted Honey pot Reviewer** can actively bait and penalize colluding generators (-211.9 Elo collapse), our work shifts the asymmetry in favor of tournament operators and safety auditors.

## LLM Usage Statement

### B.1 Exact Replication Commands and Seed Schedule

All experiments can be reproduced using `modal_vllm_server.py` (hosted on a single Modal L40S GPU) and `epistemic_cartel_tournament.py`:

1. **Seed Schedule:** Core conditions use  $\text{run\_seed} = 2026 + \text{cond\_idx} * 1000 + \text{s\_idx} * 101$  for `cond_idx` in  $\{0, 1, 2\}$  and `s_idx` in  $\{0, 1, 2, 3, 4\}$ . Extension Experiments A, B, and C use deterministic base seeds  $5000 + \text{s\_idx} * 113$ ,  $7000 + \text{s\_idx} * 97$ , and  $9000 + \text{s\_idx} * 131$ .
2. **Command Line:** `python3 epistemic_cartel_tournament.py --mode llm --rounds 30 --seeds 5 --suite full_championship --plot`

### B.2 LLM Usage Statement

In accordance with Apart Research Sprint guidelines, we disclose that LLM assistance (Gemini) was used during the sprint to help architect the `epistemic_cartel_tournament.py` and `modal_vllm_server.py` codebase, debug Python 3.14 / NumPy sandbox compatibility, and assist in structuring and formatting this report. All live experiments (1,350 generated scripts and 8,100 pairwise reviews) were executed by the author on Modal GPUs (Qwen2.5-14B-Instruct-AWQ and Meta-Llama-3.1-8B-Instruct), and all statistical claims, p-values, and metrics were independently verified against `tournament_summary.json`.