

# Can LLMs Discover Novel Economic Theories?\*

Alejandro Lopez-Lira<sup>†</sup>

Sina Seyfi<sup>‡</sup>

Yuehua Tang<sup>§</sup>

University of Florida

Wilfrid Laurier University

University of Florida

September 14, 2026

## Abstract

We develop AutoTheory, a system that uses large language models and evolutionary search to generate, derive, code, calibrate, and verify economic theories. To promote diversity, AutoTheory varies both the LLM and the expert perspective used to generate ideas, and improves promising theories by revising them or combining elements from different theories. We apply the system to three asset-pricing puzzles: the scaling of price multipliers, the equity term structure, and the currency premium puzzle. For the price-multiplier puzzle, AutoTheory finds two distinct verified equilibrium mechanisms; for the other two puzzles, it identifies promising candidates while flagging apparent solutions that rely on target-specific calibration parameters. The results show that LLMs can systematically search over economic theories, generate new quantitative mechanisms, and help distinguish genuine theoretical explanations from sophisticated curve fitting.

**Keywords:** LLMs, theory generation, evolutionary algorithms, asset pricing, price multipliers, automated discovery

---

\*The authors have no financial disclosures or conflicts of interest.

<sup>†</sup>alejandro.lopez-lira@warrington.ufl.edu

<sup>‡</sup>sseyfi@wlu.ca

<sup>§</sup>yuehua.tang@warrington.ufl.edu

# 1 Introduction

In July 2026, an artificial-intelligence system announced a counterexample disproving the Jacobian conjecture, a problem that had been open for decades. Until recently, progress in AI mathematics meant solving hard problems whose answers were already known: mathematics competitions, graduate-level exercises, curated benchmarks. That frontier has now shifted to research itself—longstanding open problems, new proofs and counterexamples, results that pass expert review. AI is moving from *learning* mathematics to *adding* to it. Mathematics moved first for a reason: a new mathematical result comes with a decisive, verifiable test. This paper asks whether the same paradigm can extend to economic theory, where models of human behavior admit no such decisive test.

The question is the natural third step in a progression. Large language models (LLMs) can *predict*: sentiment extracted from news headlines forecasts stock returns out of sample, outperforming traditional text methods (Lopez-Lira and Tang, 2023). LLMs can *trade*: LLM agents implement and maintain coherent strategies—value investing, momentum, market making—in simulated markets populated by several kinds of traders (Lopez-Lira, 2025). Whether LLMs can *theorize* is harder than either: it requires proposing an economic mechanism, deriving its implications correctly, and fitting the result to data. The progression runs from pattern recognition through strategy execution to theory generation—and it raises the question we address here: can the generation of economic theories be formalized and automated?

We address this question by building a *general-purpose system* for LLM-based theory discovery in economics. The system is puzzle-agnostic: it takes as input a specification of an empirical puzzle (target values, baseline model, calibration parameters) and outputs candidate theories that are economically plausible, parsimonious, and fit the data. It accompanies these theories with mathematical formulations, executable code, optimized parameters, and simulated referee assessments. The system makes theory search scalable: it generates many candidate mechanisms, imposes feasibility and internal-consistency

checks, disciplines them using empirical targets, and retains the strongest proposals. Numerical targets (observed empirical moments the theory must match) serve as an objective, LLM-independent quality filter. A candidate theory either reproduces the data or it does not, regardless of how persuasively the LLM describes it. This numerical anchor is what makes automated theory search viable: it grounds the evaluation in data rather than in the LLM’s own judgment. The checks are the substance of the design: without them, the search learns to *look* successful without actually solving the puzzle.

Why should LLMs be able to discover novel economic theories? A classic view of creativity and scientific discovery is that progress emerges through a generate–test–retain process: produce many ideas, evaluate them, and keep the best (Campbell, 1960). Simon-ton (1997) likewise argues that creative success is partly a numbers game, with higher output increasing the likelihood of major contributions. LLMs fit naturally into this framework because they can cheaply generate many diverse candidate theories by recombining existing concepts and mechanisms in new ways. Repeated sampling from an LLM lets researchers search more broadly over plausible explanations; if useful theories occur with non-zero probability, more draws increase the chance of finding one. Our system implements this generate–evaluate–select logic at scale.

For economists, automating this creative search stage changes the production function of theory. Instead of relying on a small number of hand-crafted mechanisms tailored to a single puzzle, researchers can systematically explore a broader model space, record failures as well as successes, and compare competing explanations using common quantitative and qualitative criteria. In practice, economists often learn by surprise: a regularity appears in the data, the current model does not explain it, and researchers iteratively revise mechanisms until a coherent explanation emerges. Heckman and Singer (2017) describe this as an abductive process. LLMs can support this process as a scalable engine for proposing candidate mechanisms that humans can scrutinize.

Methodologically, we build a puzzle-agnostic system for theory search. The pipeline

starts from a puzzle specification and generates candidate theories with LLM-guided evolution, randomizing the proposing model across several LLMs (including OpenAI’s open-weight gpt-oss-120b (OpenAI et al., 2025) and GPT-5.6-class models) while holding the verification stages fixed. Each candidate passes through the current *integrity chain*: a proposal-level gate that rejects theories which assert rather than derive their headline result; symbolic derivation of every pricing formula, with target quantities computed by harness-owned operators the model cannot re-implement; an independent mathematical audit; translation through structured pseudocode into executable code, with reviews and deterministic checks at each step; calibration and optimization against the empirical targets; deterministic reconciliation that the scored code agrees with the verified derivation; and an output-sensitivity scan that flags private dials, pinned targets, and inert parameters. A scorer grades each model on quantitative fit, economic plausibility, and parsimony, and a panel of referee LLM calls produces the final score and feedback. Integrity is therefore a property of the current executable pipeline: model-supplied proxies, hard-coded targets, derivation–code mismatches, and parameters dedicated to a single fitted target are rejected or explicitly flagged before economic evaluation.

To promote diversity, each run draws a random combination of expert personas from 20 methodological perspectives (e.g., bayesian, behavioral, physicist, minimalist). This structure keeps creativity and discipline together: broad generation at the front, hard filters and transparent evaluation at the back. Unlike recent AI-discovery systems in mathematics and the natural sciences, which optimize over fully formal objects and verifiable objectives, our setting involves subjective models of human behavior; our contribution is to make this search disciplined and reproducible despite that ambiguity.

We opt for three different strategies for the LLM-based theory generation. These three strategies are *random* (propose a mechanism from scratch), *mutate* (modify a previously successful theory, guided by its referee report), and *crossover* (recombine elements of two successful theories). A broad literature motivates this design: impactful discovery often

comes not from wholly unprecedented ideas but from unusual recombinations of existing knowledge, methods, or data (Fleming, 2001; Nerkar, 2003; Uzzi et al., 2013; Fortunato et al., 2018; Teplitskiy et al., 2022; Park et al., 2023; Yu and Romero, 2024). Random search preserves exploration; mutate and crossover let the system exploit and recombine partial insights discovered earlier in the search.

The price-multiplier puzzle illustrates the system. Gabaix and Koijen (2023) define the price multiplier  $M$  as how much an asset’s price changes when demand shifts by one dollar. Li and Lin (2026) find that the price multiplier is not constant across levels of aggregation. They show that the price multiplier is larger at the style-level than at the idiosyncratic level, and that the ratio of style-level to idiosyncratic price multipliers observed in data ( $M_{\text{style}}/M_{\text{idio}} \approx 3.6$ ) differs dramatically from the prediction of the standard CARA-normal model ( $\approx 14.5$ ). This factor-of-four gap is economically important and has a clear quantitative target, making it suitable for automated evaluation.

For the price-multiplier puzzle, we run the fully verified system for 1,000 iterations (523 passing every gate) and hand-audit the frontier. Two genuinely distinct two-parameter equilibrium mechanisms tie at the top referee score of 74/100: an *inventory-cost floor*, in which a per-position resource cost enters all multipliers equally and compresses the style-to-idiosyncratic ratio toward the data (and implies an over-identifying restriction that holds in the data to within about 5%); and *concave risk aggregation*, in which shared-risk pricing grows with group breadth at a fitted concave rate rather than linearly. Both models’ reported multipliers are computed by the harness as the comparative static of each model’s own declared equilibrium system, both reproduce their outputs exactly, and the referee panel holds both at “major revision” for the same articulated reason—the aggregation structure that does the work is imposed rather than derived from primitives—which we regard as an automatically generated statement of what a full resolution still requires. We run the same recipe on two further puzzles, with instructive results. On a currency premium puzzle (Hassan et al., 2026), the current integrity analysis classifies

the top models (62–68/100 across 1,000 runs) as fitting the one non-harness-owned target with private dials. On the equity term structure (van Binsbergen et al., 2012), where short-term dividend strips have roughly twice the Sharpe ratio of the market, the flagship experiment is complete (1,000 invocations, 288 passing every gate); the detector splits its frontier cleanly—five of the top ten models, including the raw best at 64/100, fit the conditional Sharpe targets with a dial, while the best clean model (62/100) produces the conditional pattern from its mechanism and posts an honest miss on the market return. Under the current standard, neither puzzle is yet solved.

The remainder of the paper proceeds as follows. Section 3 presents a simple model of theory discovery under parallel LLM search. Section 4 describes the architecture of our LLM-based theory-discovery system. Section 5 reports our experimental analysis for both puzzles and compares the results. Section 6 discusses capabilities, limitations, and implications of LLM-based theory discovery. Section 7 concludes.

## 2 Related Literature

Our work sits at the intersection of three literatures: AI-assisted scientific discovery, psychological and formal models of creativity, and demand-based asset pricing. We contribute to the first by demonstrating LLM-based theory generation in economics; to the second by implementing a formal generator–selector loop consistent with blind variation and selective retention; and to the third by producing theoretical mechanisms that explain empirical puzzles in asset pricing.

### 2.1 Creativity as Variation and Selection

A longstanding way to formalize creativity in science and mathematics is as a generate–test–retain process: generate many candidate ideas (variation), evaluate them (selection), and keep or elaborate the best (retention). Campbell (1960) articulates this in the “blind

variation and selective retention” (BVSR) model of creative thought: wide variation is produced by trial-and-error or fluency of ideas, and a critical or selective function decides which variants are retained. Amabile (1983) offers a complementary componential view: creativity depends on domain-relevant skills, creativity-relevant processes, and task motivation within a supportive environment, a framing that aligns with treating the LLM as a variation engine operating in a structured evaluation environment. Empirically, creative output behaves like constrained stochastic behavior: Simonton (1997) shows that hits scale with attempts (an “equal-odds” style regularity) and provides a predictive model of career trajectories and landmarks. In technological search, Fleming (2001) shows that recombinant exploration (novel combinations of existing components) increases variance in outcomes: most combinations fail, but occasional combinations generate large advances. This pattern reinforces the intuition that broad exploration can expand the space of viable candidates.

Evidence on scientific practice then highlights a selection bottleneck. Foster et al. (2015) document an “essential tension” in scientists’ research strategies: innovative directions are underused relative to their potential impact because rewards are uncertain and delayed. Wang et al. (2017) show that bibliometric indicators are biased against novelty: highly novel papers receive delayed recognition and are systematically disadvantaged by short-term citation metrics despite often achieving top long-term impact. Together, these findings motivate systematic exploration at scale: when human and institutional incentives favor tradition, automated generation can broaden the set of candidate theories that receive serious evaluation. The standard definition of creativity in psychology requires both originality and effectiveness: an idea or product must be novel and also useful or appropriate (Runco and Jaeger, 2012). We adopt this criterion by requiring candidate theories to be novel relative to a reference corpus and to pass verification (e.g., consistency checks and empirical targets). Our pipeline instantiates BVSR with the LLM as the variation engine and automated evaluators (code verification, calibration, referee) as the

selection function.

## 2.2 AI-Assisted Scientific Discovery

The use of AI systems to accelerate scientific discovery has advanced dramatically in recent years. Jumper et al. (2021) demonstrate that deep learning can solve the protein structure prediction problem, achieving atomic-level accuracy on a challenge that had resisted solution for over 50 years. AlphaFold shows that AI can make genuine scientific contributions in domains requiring complex pattern recognition.

More recently, LLM-based (or LLM-component) systems produce *new* mathematical results or formally verified proofs when paired with evaluation. Romera-Paredes et al. (2024) pair a pretrained LLM with an automated evaluator in an evolutionary loop and surpass prior best-known results in combinatorial problems (e.g., new cap set constructions). Trinh et al. (2024) present a geometry theorem-proving system that synthesizes theorems and proofs at Olympiad level without human demonstrations. Hubert et al. (2025) describe a reinforcement-learning and language-model system that finds formal proofs in Lean and achieves medal-level performance. For scientific discovery beyond mathematics, Boiko et al. (2023) present an LLM-driven agent that designs, plans, and executes complex chemical experiments using tools and automation; Luo et al. (2025) show that LLMs can surpass human experts in predicting experimental outcomes in neuroscience, supporting a role for LLMs in hypothesis generation and discovery workflows. These demonstrations instantiate the BVSr loop: LLM as variation engine, verifier or experiment as selector, archive or population as retention.

An emerging economics literature has begun to document closely related LLM use cases. On measurement and text-based inference, recent work develops and validates LLM pipelines for coding qualitative content and constructing large-scale research measures (Ludwig et al., 2025; Asirvatham et al., 2026; Hansen et al., 2023; Ottonello et al., 2024; Goetzmann et al., 2024; Galiani et al., 2026). On research production and scientific

workflow, studies show how LLMs can automate parts of economic research and, when combined with external tools or multi-agent structures, execute richer economic-analysis tasks (Korinek, 2023; Korinek, 2024; Korinek, 2025; Tranchero et al., 2024). A complementary line studies behavior, bias, and deployment: LLM-based survey instruments, adoption and use patterns, behavioral-bias diagnostics, and field decision-support interventions (Wu et al., 2025; Chatterji et al., 2025; Bini et al., 2026; Horton, 2023; Abaluck et al., 2026; Jabarian and Imas, 2025). Collectively, these papers show broad feasibility for LLM-assisted economic inquiry while also highlighting concerns about validation, bias, and context dependence.

Novy-Marx and Velikov (2026) demonstrate that LLMs can generate hundreds of complete finance papers with distinct theoretical justifications for data-mined signals, and caution that generating post-hoc theoretical justifications for data-mined patterns can industrialize hypothesizing after results are known (HARKing). HARKing is fundamentally a concern about statistical inference: presenting post-hoc hypotheses as ex-ante predictions inflates false-positive rates. Our setting is different. We start from a well-established quantitative failure of a workhorse structural model, not from a data-mined anomaly, and search for economic mechanisms that resolve it. This research design mirrors canonical theory papers that target known puzzles; for example, the equity premium puzzle motivates both Campbell and Cochrane (1999) and Bansal and Yaron (2004). Our system automates the search over candidate mechanisms while keeping the process fully transparent: all candidates are reported, not just the best.

AlphaEvolve (Novikov et al., 2025) extends this paradigm to general-purpose algorithm discovery. Using an ensemble of Gemini models with evolutionary search, AlphaEvolve discovers algorithms for matrix multiplication that improve upon Strassen’s 1969 result, optimizes Google’s data center scheduling (recovering 0.7% of worldwide compute resources), and accelerates AI training kernels. A parallel literature explores machine learning for hypothesis generation in social science: Ludwig and Mullainathan

(2024) propose using ML algorithms to surface patterns in data that human researchers might miss, generating novel hypotheses about human behavior.

Our work differs from these predecessors in important ways. Unlike AlphaFold, which operates on well-defined physical constraints, economic theory requires modeling human behavior with inherent uncertainty about “ground truth.” Unlike FunSearch, AlphaGeometry, AlphaProof, and AlphaEvolve, which optimize for formally verifiable objectives (proofs, algorithm performance), our evaluation combines quantitative fit with subjective assessments of economic plausibility. And unlike ML-based hypothesis generation, we generate complete theoretical models with analytical pricing expressions, not just statistical patterns. LLMs can generate theories with new mechanisms, mathematical formulations, and testable predictions. The evaluation pipeline applies round-trip consistency verification, parameter optimization, plausibility scoring, and simulated peer review to filter LLM proposals into calibrated economic theories.

## 2.3 The Price Multiplier Literature

Price multipliers quantify how much an asset’s market value changes when demand shifts by one dollar. Understanding these multipliers is central to demand-based asset pricing.

Gabaix and Koijen (2023) propose the “inelastic markets hypothesis,” finding that every \$1 invested in the aggregate market increases market value by approximately \$5, far larger than standard models predict. Using granular instrumental variables, they document price multipliers in the range of 3 to 8 for market-level flows, while individual stock multipliers are closer to 1. This aggregate-versus-individual gap suggests that diversifiable risk is priced differently than systematic risk.

Li and Lin (2026) provide the direct motivation for our puzzle. They decompose demand into style-level and idiosyncratic components and estimate separate price multipliers for each. They find that price multipliers form a continuum: approximately 1.6 at

the idiosyncratic level, 2.5 at the granular style level, and 5.9 at the coarse style level. They show that the standard CARA-normal model *qualitatively* predicts this ordering (higher multipliers for less diversifiable risk) but *quantitatively* fails: the model predicts  $M_{\text{style}}/M_{\text{idio}} \approx 14.5$ , while the data show  $\approx 3.6$ .

This quantitative gap, a factor of four, is our target puzzle. In a subsequent revision (January 2026, posted after the release of the LLM used in our experiments), Li and Lin (2026) add a back-of-the-envelope calibration showing that when only a fraction ( $\approx 29\%$ ) of investors actively provide liquidity, risk aversion alone can generate multipliers of the correct order of magnitude. However, treating their point estimates literally “would be heroic,” and they do not provide a fully specified equilibrium model. The quantitative discrepancy thus remains an open target for theory.

Related work on market microstructure provides foundations for understanding price impact. Kyle (1985) derives equilibrium price impact from informed trading, while Glosten and Milgrom (1985) model bid-ask spreads with heterogeneous information. Hasbrouck (2009) develops methods for estimating effective trading costs from daily data. These papers focus primarily on information-driven price impact; our puzzle concerns risk-based price impact where the Li and Lin (2026) evidence suggests information plays a limited role.

The contribution to this literature is methodological rather than empirical. LLM-based evolutionary search, under full verification, generates distinct theoretical mechanisms that quantitatively match the empirical puzzle—an inventory-cost floor on price impact and concave risk aggregation across breadth—each with two free parameters, testable implications, and, in the first case, an over-identifying restriction that holds in the published estimates. Multiple mechanisms can fit the same data; the system generates and adversarially verifies candidate explanations at a scale that would be infeasible for human researchers alone.

### 3 A Model of Theory Discovery

This section formalizes theory discovery as a stochastic search problem under verification constraints—a small economic model of the discovery process itself. The framework reduces to three primitives: the system must *generate* a genuine solution with positive probability, *distinguish* a genuine solution from a plausible fake with positive probability, and keep its draws *effectively uncorrelated* so that scale compounds. Read as a production function for verified discoveries, expected output is  $n_{\text{eff}} p_1 p_2$ : the inputs are complements—the product form means the marginal value of scale is zero when either generation or discrimination is absent—and each input maps to a distinct layer of the pipeline that the rest of the paper measures and improves.

#### 3.1 Setup

Let  $\mathcal{T}$  denote the finite but large space of candidate theories for a puzzle specification  $s$ . A candidate  $\tau \in \mathcal{T}$  can be represented as text, equations, and executable code. The pipeline maps each  $\tau$  to a quality score  $q(\tau) \in [0, 1]$  (normalized from a 0–100 scale) that aggregates implementation validity, empirical fit, economic plausibility, and parsimony. For a publication threshold  $\bar{q}$ , define the acceptable set  $\mathcal{A} = \{\tau \in \mathcal{T} : q(\tau) \geq \bar{q}\}$ .

Following the creativity literature, we distinguish effectiveness from originality. Let  $\mathcal{K}$  be a reference corpus of known mechanisms. The set of creative theories is

$$\mathcal{C} = \{\tau \in \mathcal{T} : \tau \text{ is novel relative to } \mathcal{K} \text{ and } q(\tau) \geq \bar{q}\}. \quad (1)$$

Novelty here encompasses genuinely new mechanisms or novel combinations of known building blocks, consistent with the recombinant view of creativity in Fleming (2001). This operationalizes the originality-and-effectiveness criterion in Runco and Jaeger (2012) within a fully auditable evaluation pipeline.

### 3.2 Discovery requires generation *and* discrimination

Conditional on  $s$ , the LLM induces a distribution  $P_L(\cdot \mid s)$  over  $\mathcal{T}$ . One pipeline run draws a candidate  $\tau \sim P_L(\cdot \mid s)$  and subjects it to verification. In the language of Campbell (1960), this is a generate–test–retain process: generation is stochastic variation, verification is selection, and the model registry is retention. Two primitives govern it:

$$p_1 = \Pr_{\tau \sim P_L(\cdot \mid s)}(\tau \in \mathcal{A}), \quad p_2 = \Pr(\text{verifier retains } \tau \mid \tau \in \mathcal{A}), \quad (2)$$

the probability that a draw is genuinely acceptable, and the verifier’s *power*—the probability that it recognizes a genuine solution when it sees one. Under independent draws, the probability of at least one verified discovery in  $n$  runs is

$$\Pr(\text{at least one verified discovery}) = 1 - (1 - p_1 p_2)^n, \quad (3)$$

which converges to one as  $n \rightarrow \infty$  if and only if  $p_1 p_2 > 0$ , with expected count  $n p_1 p_2$ . The product structure is the section’s central point: generation and discrimination are *jointly* necessary, and neither substitutes for the other. A generator with  $p_1 > 0$  feeding a verifier with  $p_2 = 0$  converts unlimited compute into nothing; a perfect verifier attached to a generator that never lands in  $\mathcal{A}$  has nothing to verify. Human science satisfies the same condition through referees and replication; the automated version makes both probabilities explicit design objects.

**Why scale matters in symbolic search.** Three structural features make the scaling in (3) relevant for theory generation. First, outcomes in symbolic domains are heavy tailed: a small share of candidates accounts for most value, so best-of- $n$  sampling has large returns. Second, the search space is combinatorial: even a limited library of primitives can be recombined into a very large set of coherent mechanisms. Third, sampling controls provide a direct exploration–exploitation margin at low marginal cost. Discovery rates can there-

fore rise sharply with throughput even when per-draw quality is modest—*provided* the other two primitives hold. This is the same statistical logic behind the empirical regularity that high-impact discoveries scale with attempts in stochastic models of creativity (Simonton, 1997).

### 3.3 Better verification: the false-pass side of $p_2$

Power is only half of discrimination. Suppose a draw that is not genuine is merely *plausible*—wrong in a way not visible on its face (a relabeled internal quantity, a smuggled constant, a free parameter serving as a private dial on one target)—and the verifier retains such a fake with probability  $\phi$ , its *false-pass rate*. A batch of  $n$  draws then retains  $np_1p_2$  genuine theories and  $n(1 - p_1)\phi$  fakes in expectation, so the probability that a retained candidate is genuine is

$$\Pr(\text{genuine} \mid \text{retained}) = \frac{p_1p_2}{p_1p_2 + (1 - p_1)\phi}. \quad (4)$$

Scale cancels from (4):  $n$  multiplies both terms, so the informativeness of “the search found something” is governed entirely by the verifier’s likelihood ratio  $p_2/\phi$  against the prior odds  $p_1/(1 - p_1)$ . Because genuine hits are rare ( $p_1$  small), even a modestly leaky verifier makes the retained set mostly fakes, no matter how large  $n$  grows.

Adaptive search sharpens the problem, because  $\phi$  is not a fixed constant. An evolutionary loop that selects parents on the verifier’s own output concentrates sampling exactly where the verifier leaks, so the *effective* false-pass rate rises over a run: the search does not merely encounter the verifier’s blind spots, it finds them. The current system therefore rests integrity on *deterministic* checks—symbolic derivation with harness-owned operators, code-derivation reconciliation, AST-based constraints, and the private-dial detector—for which  $\phi \approx 0$  on the dimensions they cover, rather than on LLM judgment alone. The referee’s economic judgment ranks retained theories, but the deterministic

components govern whether the reported quantitative objects are faithful to the stated model. Section 4 describes this current division of labor.

### 3.4 The trick is uncorrelated draws

Equation (3) assumes independence, and in practice draws from a single LLM prompt are far from independent: the model’s training distribution, prompt structure, and decoding temperature induce positive correlation across candidates, so an effective sample size  $n_{\text{eff}} < n$  replaces  $n$ . Since generation is cheap, correlation—not compute—is the binding constraint on scale, and most of the pipeline’s diversity machinery exists to keep  $n_{\text{eff}}$  growing with  $n$ .

The persona mechanism attacks within-model correlation. Let  $\Pi = \{\pi_1, \dots, \pi_K\}$  be a set of  $K$  expert persona configurations. We define 20 base personas (methodological perspectives such as “bayesian,” “physicist,” “minimalist”); each run either uses a default perspective (with probability 0.2) or randomly samples a combination of one to three base personas (with probability 0.8), yielding  $K = \binom{20}{1} + \binom{20}{2} + \binom{20}{3} + 1 = 1,351$  distinct configurations. Conditioning the generator on configuration  $\pi_k$  shifts its distribution to  $P_L(\cdot \mid s, \pi_k)$ , so the unconditional generator is the mixture

$$P_L(\cdot \mid s) = \sum_{k=1}^K w_k P_L(\cdot \mid s, \pi_k). \quad (5)$$

When the component distributions have limited overlap, draws from different components are approximately independent even if draws within a component are correlated, and  $n_{\text{eff}}$  scales with the number of distinct components explored. The flagship experiments extend the same logic across model families by randomizing the *proposer* over five LLMs per run: different models’ training distributions are different mixture components.

Correlation also enters through the evolutionary loop itself. The current policy draws parents from the full scored population with probability proportional to score squared

and requires cross-lineage crossover whenever distinct lineages exist. In the price-multiplier flagship, the leading lineage accounts for 15% of scored descendants, 82% of parent selections come from outside it, and the top ten span five distinct model fingerprints. Fingerprinting supplies the honest accounting throughout: what matters for  $n_{\text{eff}}$  is distinct *ideas* explored, not the raw count of runs or model names.

### 3.5 Optimization in the space of economic models

The evolutionary layer is best understood as optimization *directly in the space of economic models*. The individuals are theories; mutation and crossover are moves in  $\mathcal{T}$  (modify one mechanism, recombine two); the referee panel’s written reports are the gradient signal that directs the next move. This raises the effective generation probability around discovered basins: conditional on a strong parent, the success rate of a directed draw exceeds that of a fresh one, and the experiments bear this out—on the price multiplier both top models are generation-4 and generation-5 mutations and crossover attains the highest mean score, while on the dividend strips directed draws outscore fresh ones by about eight referee points and supply nine of the top ten models.

The same property is why verification cannot be relaxed as the search improves: an optimizer in model space climbs whatever gradient the verifier defines. Where every target’s definition is harness-owned, refinement produces better equilibrium models; where a target is not mechanically owned, the current sensitivity scan tests whether fit comes from a private dial on that target (Section 5). Optimization in model space and the integrity condition of Section 3.3 are therefore complements, not separate features: the former makes the search powerful, the latter makes its output meaningful.

### 3.6 Economic comparison with human-only search

Suppose a human researcher draws from  $P_H$  at expected cost  $c_H$  per attempt and success probability

$$p_H^A = \Pr_{\tau \sim P_H} (\tau \in \mathcal{A}). \quad (6)$$

The expected cost per acceptable theory is  $c_H/p_H^A$ . For an LLM system with per-attempt cost  $c_L$  and acceptable-hit probability  $p_1$ , the analogous object is  $c_L/p_1$ . LLM search has a cost advantage whenever

$$\frac{c_L}{p_1} < \frac{c_H}{p_H^A}. \quad (7)$$

This condition can hold even if  $p_1 < p_H^A$ , because  $c_L$  is orders of magnitude smaller and generation can be parallelized.

The same cost asymmetry governs the *screening* side, where the scarce input is expert attention. A false positive—a candidate that passes automated verification but is not actually useful—must still be rejected by a human. A brief human screen of one candidate takes on the order of ten minutes; at \$120 per hour that is roughly \$20 per candidate. A three-referee LLM panel costs on the order of \$0.01 per candidate, so the price of one human screen buys about 2,000 independent panels. The design question is therefore not whether LLM referees match a human expert one for one—they do not—but how to allocate a fixed budget of expert minutes: automated screening exists to ensure that the candidates that do reach a human are drawn from the far right tail of the verified distribution. An analogous comparison can be made for creative hits by replacing  $\mathcal{A}$  with  $\mathcal{C}$  for both humans and LLMs; the qualitative point is the same, but acceptable theories are the more directly observable object in our pipeline. In Section 5, we proxy novelty using cosine distance between proposal embeddings.

### 3.7 Search policy

The framework implies a two-layer policy that the pipeline implements. A base layer of independent random draws (personas and randomized proposers doing the decorrelation) maps the space and founds lineages—every lineage’s generation-0 ancestor is such a draw. A directed layer (mutation, crossover, score-squared parent selection over the full population, and cross-lineage mating) climbs the discovered basins, with ongoing random mass preserved to prevent premature concentration. If  $p_r$  is the success rate of random generation and  $p_m$  the conditional success rate of directed draws around strong parents, expected output decomposes as  $N_r p_r + N_m p_m$ , and the observed pattern across puzzles is  $p_m > p_r$  wherever the parent pool clears a quality floor. The current lineage and fingerprint diagnostics show that directed search remains distributed across distinct founding proposals.

The following sections describe the pipeline that implements this logic: how candidate theories are generated and translated into code, how the verification chain is enforced, and how strategy selection is applied across a large number of runs.

## 4 System Design

Our system executes a sequence of stages organized into three phases: *generation*, *verification*, and *evaluation*. Each stage saves its output incrementally, so partial results are preserved even if later stages fail. The system is puzzle-agnostic: each puzzle defines its own target outputs, calibration inputs, and prompts through a modular interface, allowing the same pipeline to be applied to different empirical puzzles without modification.

The current architecture separates numerical fit from definitional integrity. A numerical optimizer can force a model’s *outputs* toward targets, but it cannot by itself verify that the code computes the quantity the target *defines*. The pipeline therefore interleaves the generative stages with an *integrity chain* of gates—most of them deterministic—that verify

provenance at every translation step (proposal to symbolic derivation to code to reported number) and *reject* runs that fail, rather than scoring them. Sections below describe each stage and its gate; Appendix C.3 summarizes the retry and rejection policy.

A back-of-the-envelope calculation shows why the staged-and-gated structure, rather than a single long prompt, is load-bearing. Producing a theory end to end chains  $K$  linked tasks (propose, derive, translate, code, calibrate, ...); if each succeeds with probability  $s$ , one monolithic attempt succeeds with probability  $s^K$ —for  $K = 10$  and  $s = 0.8$ , about 10.7%. Chaining alone does not help: the gain comes from *checkable intermediate outputs*. When each stage’s failure is detectable and the system can retry the failed stage up to three times, the success probability becomes  $[1 - (1 - s)^3]^K \approx 92.3\%$  for the same values, because the system repairs the stage that failed instead of restarting the chain. The same logic applies to judgment: if one referee is right with probability 0.8, the majority of three independent referees is right with probability 89.6%. These calculations are illustrative—they assume independent errors and fresh attempts—but they capture why the pipeline invests in verification at every handoff rather than in longer prompts.

## 4.1 Generation Phase

**Problem Analysis with Expert Personas.** Given a puzzle definition, an LLM analyzes the problem and identifies the empirical facts that need explaining. The *problem analyzer*, an LLM equipped with the problem description and baseline model, describes the empirical facts, explains why the baseline model fails, and highlights both promising directions for new theoretical explanations and likely dead ends. The *problem analyzer* also determines the constraints that any successful theory must satisfy and highlights one unconventional angle worth exploring.

To promote diversity in the generated theories, each run randomly assigns an *expert persona* to the problem analyzer. We define 20 personas spanning different methodological perspectives: *mathematician* (fixed-point theorems, duality), *bayesian* (priors, posteri-

ors, value of information), *behavioral* (bounded rationality, heuristics), *physicist* (conservation laws, symmetries), *minimalist* (fewest assumptions possible), *contrarian* (challenges conventional wisdom), and others.<sup>1</sup> With probability 0.2, the run uses a default perspective; with probability 0.8, it randomly samples one to three personas and combines their perspectives by including all selected persona descriptions in the LLM’s system prompt simultaneously. This mechanism ensures that the system explores qualitatively different theoretical directions across runs, reducing the tendency of LLMs to converge on the same class of solutions. See Appendix A.1 for the prompts used in the problem analyzer.

**Theory Evolution.** The *theory evolution* stage receives the problem description and baseline model from the problem analyzer. Inspired by the output of the problem analyzer, it proposes a new theoretical mechanism that potentially explains the empirical facts.

For *theory evolution*, we implement three strategies. The first is *Random*, which generates a new theory from scratch. Many scientific discoveries evolve from combinations of previous theories (Uzzi et al., 2013), so we add two strategies that leverage existing theories. Once successful theories exist, the *theory evolution* LLM uses them as parents to generate new theories. One option is *Mutate*, which modifies an existing successful theory. The mutate prompt includes the parent’s formulas, parameters, calibration results, and the full referee report (strengths, weaknesses, suggestions), directing the LLM to address at least one identified weakness. Examples of mutation include simplifying the parent model, changing parameter mappings, or adding interactions and new effects.

The third strategy is *Crossover*, which combines mechanisms from two successful theories and creates a hybrid model using the best elements of both. The crossover prompt directs the LLM to aim for fewer total parameters than the sum of the two parents by finding overlap and eliminating redundancy.

---

<sup>1</sup>The full list of personas includes: default, mathematician, game theorist, information theorist, decision theorist, bayesian, econometrician, robust statistician, macro theorist, micro theorist, financial economist, behavioral, institutional, monetary economist, network theorist, dynamic programmer, minimalist, contrarian, physicist, and pragmatist.

Once at least one model has completed the full pipeline, strategies are selected probabilistically: 35% *random*, 35% *mutate*, 30% *crossover*. Parents are drawn from the full population of scored models with selection probability proportional to the *square* of the referee score—no rank cutoff and no eligibility floor—so that a fresh theoretical family scoring within referee noise of the leader retains a meaningful chance of being bred, while selection pressure still concentrates on the frontier. For crossover, the second parent is drawn from a different *lineage* (chain of ancestry back to a founding random proposal) than the first whenever one exists, so that crossover combines two families rather than recombining one.

The output of the *theory evolution* stage is a new theoretical mechanism that potentially explains the puzzle. See Appendix A.2 for the prompts used in theory evolution.

**Gate: Proposal Screening.** Before any derivation work begins, an LLM judge panel screens whether the raw proposal *derives* its headline result from the required solution concept rather than asserting or arranging it. For example, a currency proposal must pin the exchange rate down in equilibrium rather than postulate a free drift exactly where the target moment lives. Each puzzle declares the solution concept a proposal must genuinely deliver (e.g., a determinate market-clearing equilibrium in which the reported quantity is endogenous); five independent judge passes vote, and a strict majority is required to flag. Flagged proposals are sent back to the theory-evolution stage with the judges’ rationale (the same strategy and parents are retried, so the gate does not distort the strategy mix); a proposal still flagged after the retry budget is rejected outright. Puzzles whose reported quantities are verified mechanically downstream can opt out of this gate.

## 4.2 Verification Phase

**Symbolic Derivation (SymPy Agent).** An LLM translates the proposal’s derivation into an executable SymPy program, which the system runs to obtain closed-form expressions for every target moment. This turns the derivation itself into an artifact that can be exe-

cuted, checked, and diffed, rather than prose algebra taken on trust.

Crucially, the derivation may not hand-write the quantities the puzzle is about. The harness owns a small library of symbolic operators—for example, an operator that computes risk-free rates and Hansen–Jagannathan bounds from a candidate log stochastic discount factor, an operator for the moments of a realized log return, and an operator that computes an equilibrium price response as the implicit-function-theorem comparative static of the model’s declared equilibrium system. Each puzzle declares which operators a valid derivation *must* invoke; the system tracks real invocations and retries derivations that skip one. For declared *owned outputs*, the printed quantity must additionally be symbolically equal to a value the operator returned. The equilibrium-response operator rejects degenerate declared systems with fewer than two equations or with unknowns pinned directly by the shock.

**Gate: Transcription Check.** A separate check verifies that the SymPy derivation faithfully transcribed the proposal: an LLM extracts (symbol, quotation) claims from the proposal, and a deterministic verifier confirms whether any structural term the proposal specifies was silently dropped from the derivation (declared, quoted, and used in no expression). Confirmed drops trigger re-derivation; persistent drops reject the run.

**Mathematical Audit.** An independent LLM call, operating at low temperature, audits every derivation step in the (now SymPy-verified) proposal. The auditor checks that each equation follows rigorously from the previous one, that matrix algebra and first-order conditions are correct, and that market-clearing conditions hold, returning a PASS or FAIL verdict with detailed feedback. If the audit fails, the system sends the proposal together with the auditor’s feedback to another LLM call that attempts to fix the mathematical errors while preserving the economic mechanism, then re-derives and re-audits. This fix-and-reaudit cycle repeats up to nine times; across 7,369 audited runs, 15% of proposals pass on attempts 4–10 and score no worse than early passers, so the deep retry budget rescues models more cheaply than fresh runs would. Proposals that exhaust the audit

budget are discarded.

**Pseudocode Generation and Review.** Rather than translating proposals directly into Python, we introduce an intermediate pseudocode stage that forces the LLM to think through the complete derivation chain before coding. The *pseudocode agent* LLM converts the audited proposal into structured pseudocode organized into four sections: parameters (with types and bounds), derivation steps (from economic primitives to pricing equations), output computation (mapping derived quantities to target moments), and conditional moments (if the puzzle requires state-dependent predictions). A separate *pseudocode reviewer* LLM then verifies that the pseudocode faithfully captures the proposal’s economic mechanism, that all key equations are present, and that outputs are derived from the model rather than hardcoded; if it identifies issues, the pseudocode agent regenerates with the reviewer’s feedback.

**Code Translation, Deterministic Checks, and Review.** The *coding agent* LLM translates the pseudocode into executable Python. It receives the pseudocode specification, the original proposal, and—critically—the verified SymPy formulas, which the system additionally compiles into self-contained Python helper functions that the generated code is instructed to call rather than re-deriving the moments by hand. The verified formula block is appended to the generated code so that the verified versions prevail even if the LLM redefines them. The coding agent uses a nested retry strategy: up to two fresh starts, each with up to three retries, for a maximum of six attempts.

Every generated file must pass a set of deterministic, AST-based checks before it is accepted. A *state-realization gate* rejects code that exposes a time- $t$  state or regime realization as an optimizable parameter (which would let the optimizer tune the “data” rather than the model); state names are taken from the verified derivation’s helper signatures and from naming conventions. A *hardcoded-target check* rejects any literal equal to a target moment to three significant digits, including literals that hit a target exactly only after a time-unit conversion. Literal defaults declared through dataclass field machinery count

as literals. Finally, the free-parameter set is fixed by declaration: the calibration stage’s declared list is authoritative, so puzzle constants and unit conventions cannot re-enter as tunable parameters under alias names.

After code generation, a *code-vs-pseudocode reviewer* LLM verifies that the Python code correctly implements the pseudocode specification—parameters, derivation equations, output formulas, and conditional logic. A failed review triggers regeneration with the reviewer’s feedback; a review that still fails after retries *rejects* the run rather than passing it through. For puzzles that declare solution constraints (e.g., a closed-form requirement), a separate single-purpose LLM judge, the *constraint agent*, rejects code that reaches the targets by numerical device—root finders, numerical integration, Monte Carlo, or iterative solvers—in place of the declared solution concept.

### 4.3 Evaluation Phase

**Calibration Agent.** Once the *coding agent* generates Python code, the *calibration agent* determines parameter specifications for the subsequent optimization stage. It takes the model proposal and the generated code as input and suggests parameter bounds for optimization, default values, test values for robustness checking, and parameter types. This separation keeps calibration concerns distinct from code generation; the resulting declared free-parameter list also anchors the parsimony score and the deterministic parameter checks above. See Appendix A.4 for the prompts used in the calibration agent.

**Parameter Optimization.** The *optimizer* is a Python component (not an LLM) that: (i) executes the generated code and runs a *robustness check* at multiple parameter values (supplied by the calibration agent) to ensure the model runs without errors or invalid outputs; (ii) uses *differential evolution* to minimize the sum of relative squared errors  $L(x) = \sum_i ((y_i(x) - t_i) / t_i)^2$ , where  $x$  is the parameter vector,  $y_i(x)$  the model’s  $i$ -th output, and  $t_i$  the corresponding empirical target; and (iii) reports whether the model achieves the targets and the resulting outputs for use in scoring and the referee. See Appendix A.9 for

the objective function and optimization details.

**Gate: Code–Derivation Reconciliation.** After optimization, a deterministic reconciler substitutes the optimizer’s parameters (and perturbations of them) into both the verified SymPy formulas and the generated code, and compares each core moment numerically. This closes the remaining gap between the verified derivation and the code that was actually scored: a model whose code disagrees with its own verified formulas at the optimum is regenerated once and re-optimized, and rejected if the disagreement persists. Moments the puzzle declares as necessarily reducing to pure numbers must be compared—a moment that silently escapes comparison (for example, through a renamed symbol) fails the gate rather than passing it.

**Output-Sensitivity Scan (private-dial detector).** A final deterministic, advisory stage probes the optimized model numerically. Perturbing each free parameter (and each calibration input) around the optimum, it flags three patterns: a *private dial*—a free parameter, or a joint parameter direction found by a rank test on the output Jacobian, whose only independent effect on any fitted moment is a single target, after collapsing the puzzle’s declared output identities so that a dial cannot hide behind an accounting pass-through; a *pinned target*—a fitted moment that no free parameter moves at all, indicating it is fixed by construction rather than calibrated; and *inert parameters* that move no declared output. Its findings are shown verbatim to every referee vote alongside the code-scan warnings.

**Quantitative Scoring.** After the optimizer returns optimized parameters and model outputs, the *scorer* computes a quantitative score (0–100) for each model. The scorer is a Python component that combines three dimensions with configurable weights: *fit* (50%) measures how closely the model’s outputs match the empirical targets, using percentage errors and an exponential decay; *plausibility* (25%) checks whether the optimized parameters lie within economically reasonable bounds (e.g., risk aversion  $\gamma$  in a standard range, fractions in  $[0, 1]$ ) and penalizes out-of-bounds values; *parsimony* (25%) rewards fewer free parameters, judged relative to the number of independent moments the puzzle scores:

over-identified models (fewer parameters than fitted moments) score high, just-identified models receive an intermediate score, and under-identified models—whose perfect fit is contentless—are penalized steeply. The weighted total is reported to the referee as a diagnostic; models are ranked, and parents selected, by the referee score below. A large gap between the quantitative composite and the referee score is itself informative, flagging exact-fit models whose fit the referee correctly discounts. See Appendix A.10 for the exact formulas.

**Simulated Peer Review (Referee Panel).** The *referee* is an LLM that acts as a senior journal referee. It receives the model proposal, the quantitative scores from the scorer, the optimization results, and any warnings raised by the deterministic code scan. Before scoring, the referee performs a *mechanism audit*: for each factor, shock, or parameter that the proposal’s narrative claims drives a target moment, it checks whether that symbol actually appears in the derived formula for that moment, flagging decorative ingredients and state-dependence claims whose formulas are algebraically identical across states. It then produces a score (0–100), a recommendation (reject, major revision, minor revision, or accept), and structured feedback: strengths, weaknesses, suggestions, novelty assessment, and economic plausibility.

Because single referee calls carry sampling variance of roughly  $\pm 8$  points, the recorded referee score is the *median of three independent referee calls*; all three reports are stored and fed back to the theory-evolution stage, so a mutate prompt sees the full panel. Referee backends also differ systematically in generosity, so score comparisons across experiments are made only at a fixed referee model. The referee score is the pipeline’s ranking criterion throughout: it determines parent selection, leaderboards, and every result table in this paper.

**Auto-Save and Model Registry.** The system saves successful models (those that pass every gate and receive a referee score) to a persistent registry along with all metadata: the proposal, code, pseudocode, verified derivation, calibration bounds, optimized pa-

rameters, model outputs, quantitative scores, the full referee panel, strategy used, parent model(s), generation number, and persona. Registry records are keyed by the run’s random seed, making every model exactly reproducible from its identifier; lineage links (parent, second parent, founding proposal) are stored as identifiers, and a *fingerprint*—a hash of the model’s reported outputs and free-parameter count—identifies when two differently-named records embody the same model, which proves essential when measuring how many distinct ideas a run actually discovered. This registry serves as the population from which future runs select parents for mutation and crossover.

## 5 Experimental Analysis

We apply the system to three empirical puzzles in asset pricing, each under the same flagship recipe (randomized proposers, referee panel, score-squared parent selection). For the price-multiplier scaling puzzle we report complete results from a 1,000-run experiment whose top-scoring models survive a line-by-line hand audit. For the currency premium puzzle the full-scale 1,000-run experiment is complete and the current integrity analysis identifies a private-dial caveat in its top models. For the equity term structure (dividend strip) puzzle the flagship experiment is likewise complete at 1,000 invocations, with the deterministic dial detector applied to its full frontier. Throughout, the referee score is the pipeline’s ranking criterion, and all reported scores use the current referee extraction and integrity classification.

### 5.1 Setup

The flagship experiments randomize the *proposer* across LLMs while holding the rest of the pipeline fixed. For each run, the problem analyzer and theory-evolution stages are assigned one of five models—gpt-oss-120b (OpenAI et al., 2025), DeepSeek-V4-Flash, Tencent Hy3, Gemma-4-31B, or GPT-5.6-luna—drawn deterministically from the run’s seed,

Table 1: Model assignment by pipeline stage (flagship experiments)

| Stage  | Component                                      | Model                                                  |
|--------|------------------------------------------------|--------------------------------------------------------|
| 0      | Problem analyzer (with personas)               | randomized per run <sup>†</sup>                        |
| 1      | Theory evolution (random / mutate / crossover) | randomized per run <sup>†</sup> (same draw as Stage 0) |
| 1-gate | Proposal gate (5-vote panel)                   | GPT-5.6-luna                                           |
| 1a     | Symbolic derivation (SymPy agent)              | GPT-5.6-luna                                           |
| 1a-b   | Transcription gate                             | GPT-5.6-luna                                           |
| 1b     | Mathematical auditor                           | GPT-5.6-luna                                           |
| 2–2b   | Pseudocode agent and reviewer                  | GPT-5.6-luna                                           |
| 3–3c   | Coding agent, code reviewer, constraint agent  | GPT-5.6-luna                                           |
| 4      | Calibration agent                              | GPT-5.6-luna                                           |
| 5, 5b  | Optimizer; code–SymPy reconciler               | (Python)                                               |
| 6      | Quantitative scorer                            | (Python)                                               |
| 7      | Referee (median of 3 independent calls)        | GPT-5.6-luna                                           |

*Notes:* <sup>†</sup>Drawn per run, deterministically from the run seed, from {gpt-oss-120b, DeepSeek-V4-Flash, Ten-cent Hy3, Gemma-4-31B, GPT-5.6-luna}. All three flagship experiments use this assignment. GPT-5.6-luna is accessed through the Codex CLI, which exposes no temperature or seed control; gpt-oss-120b stages use the pipeline’s per-stage temperature settings (generative stages 0.5–0.7, verification and coding stages 0.1–0.2, referee 0.3). The proposal gate runs only on puzzles that declare a required solution concept (the price multiplier and currency puzzles); the dividend-strip variant used here is a reduced-form pricing-kernel puzzle and declares none.

so the assignment is reproducible and balanced. All remaining LLM stages (proposal gate, symbolic derivation, transcription gate, math audit, pseudocode and code agents and their reviewers, calibration, and the referee) run on GPT-5.6-luna. The recorded referee score is the median of three independent referee calls. Table 1 summarizes the assignment. Because referee generosity differs systematically across models, scores are comparable only within a fixed referee backend; we flag every cross-experiment comparison below accordingly. The optimizer and scorer are deterministic Python components.

The quantitative score for each model is  $0.50 \times \text{fit} + 0.25 \times \text{plausibility} + 0.25 \times \text{parsimony}$ , with each component on a 0–100 scale (Appendix A.10). It is reported to the referee as a diagnostic; models are ranked by the referee score.

## 5.2 Puzzle 1: Price Multiplier Scaling

### 5.2.1 The puzzle and calibration targets

**Definition of the puzzle.** The puzzle we target is the *scaling of price multipliers across aggregation levels*: how much an asset’s market value changes per dollar of demand shift depends on whether that demand is decomposed as style-level (common to many stocks), granular-style-level, or idiosyncratic (stock-specific). The price multiplier  $M$  is the elasticity of market value to demand; higher  $M$  means prices respond more to a given flow. Li and Lin (2026) estimate these multipliers from U.S. stock data by decomposing order flow into coarse style ( $3 \times 3$  size–book-to-market portfolios), granular style ( $6 \times 6$ ), and idiosyncratic components. They find that multipliers form a monotonic gradient: coarse style has the largest price impact, granular style is in between, and idiosyncratic is the smallest. Their baseline estimates are  $M_{\text{style}} \approx 5.917$  (coarse style),  $M_{6 \times 6} \approx 2.525$  (granular style), and  $M_{\text{idio}} \approx 1.635$  (idiosyncratic). The empirical ratio is

$$\left( \frac{M_{\text{style}}}{M_{\text{idio}}} \right)_{\text{data}} = \frac{5.917}{1.635} \approx 3.62. \quad (8)$$

A natural risk-sharing benchmark implies a much larger ratio. If the marginal investor has CARA–normal preferences and absorbs flow-induced inventory, the required risk premium scales with the variance of the position. A style-level flow shock hits all  $N_s$  stocks in a style in the same direction, so the variance of the absorbed position scales with  $N_s \sigma_{\text{style}}^2$ ; an idiosyncratic shock affects one stock, so the relevant variance is  $\sigma_{\text{idio}}^2$ . Under this aggregation, the benchmark predicts (see Appendix B for the derivation)

$$\frac{M_{\text{style}}}{M_{\text{idio}}} = \frac{N_s \sigma_{\text{style}}^2}{\sigma_{\text{idio}}^2}. \quad (9)$$

With plausible calibration inputs ( $N_s = 253$ ,  $\sigma_{\text{style}} = 0.104$ ,  $\sigma_{\text{idio}} = 0.435$ ), this ratio equals approximately 14.5. The *puzzle* is that the data show the correct qualitative ordering but

a far flatter scaling: the empirical ratio is about 3.6, while the benchmark predicts about 14.5, a factor-of-four gap—and the intermediate granular level pins down the *shape* of the gradient between the endpoints.

**Calibration and targets used in the experiment.** Following Li and Lin (2026), we use  $N = 2,278$  total stocks,  $S = 9$  coarse styles with  $N_s = 253$  stocks per style, and—for the granular level—36 granular styles with  $N_g = 63$  stocks each. Style volatility is  $\sigma_{\text{style}} = 0.104$  (annual) and idiosyncratic volatility is  $\sigma_{\text{idio}} = 0.435$  (annual). Three multiplier levels are fitted as independent targets:  $M_{\text{style}} = 5.917$ ,  $M_{6 \times 6} = 2.525$ , and  $M_{\text{idio}} = 1.635$ ; the ratio  $M_{\text{style}}/M_{\text{idio}} = 3.62$  is a derived diagnostic, not itself fitted (fitting the three levels forces a model to reproduce the gradient’s shape, not just its endpoints). The optimizer minimizes the sum of squared relative errors across the fitted outputs (Appendix A.9). A run is *successful* if and only if it passes every gate in Section 4—including the proposal gate, transcription gate, code review, and code–derivation reconciliation—and the optimized model reproduces the targets within tolerance.

**Executable definition of the multiplier.** For this puzzle the harness *owns the definition of the reported object*: each derivation must declare a multi-equation equilibrium system explicitly, and the three reported multipliers must equal the implicit-function-theorem comparative static  $\partial p/\partial q$  of that system, computed by the harness’s symbolic operator (Section 4.2). The current operator rejects systems with fewer than two equations and systems whose unknowns are pinned directly by the shock. A model reaches scoring only when its printed formulas and executable code reproduce the operator’s comparative statics.

## 5.2.2 Experiment design

The flagship experiment (proposer randomized over five models, GPT-5.6-luna elsewhere, referee panel of three; Table 1) executes 1,000 pipeline invocations, of which 523 (52%)

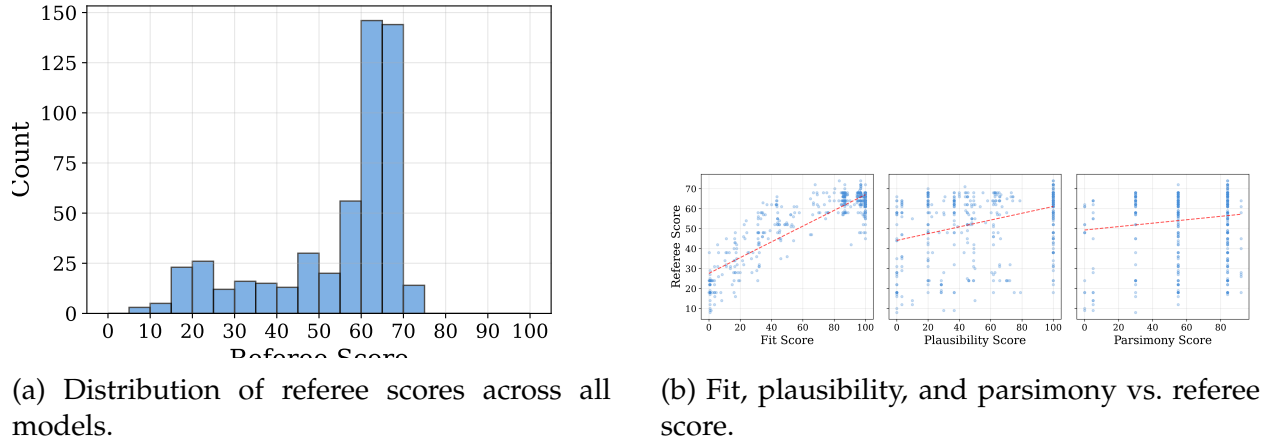


Figure 1: Score distributions of generated models (price multiplier puzzle).

*Notes:* Panel (a) reports the distribution of final referee scores (median of a three-vote panel) on a 0–100 scale. Panel (b) plots each quantitative component score (fit, plausibility, parsimony) against the referee score, with linear trend lines. The best referee score is 74, achieved by two models with distinct mechanisms.

pass every gate and enter the model registry. The largest single filter is the proposal gate, which rejects 21% of runs; the transcription gate (8%), code review (7%), and the symbolic-derivation stage (4%) account for most of the remainder (Table 13). Once at least one model completes the pipeline, each run draws a strategy (random 35%, mutate 35%, crossover 30%) and parents under the score-squared rule of Section 4.1.

The proposal gate (run by GPT-5.6-luna) censors proposers unevenly—gpt-oss-120b proposals rarely survive it (a single random-strategy survivor in this experiment)—so proposer-level statistics measure gate survival and refinement throughput, not fresh-idea quality. We therefore base the substantive conclusions on aggregate outcomes and audited frontier models.

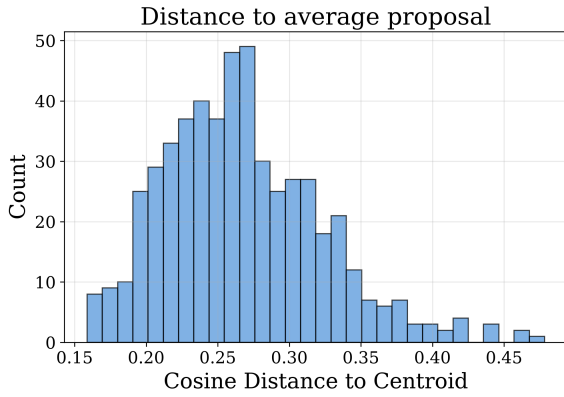
### 5.2.3 Score distributions

Figure 1 shows the distributions of referee scores and quantitative component scores.

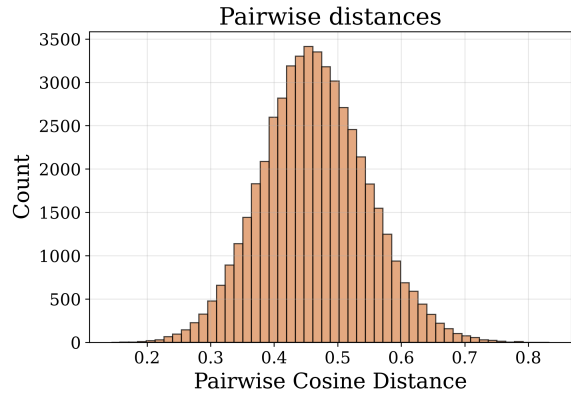
Table 2 reports summary statistics. The mean referee score is 54.6 (standard deviation 16.0, median 62); 304 models (58%) score at least 60, and none reaches 80—the GPT-5.6-luna referee holds the entire top of the distribution at “major revision,” for reasons the

Table 2: Summary Statistics (Price Multiplier)

| Statistic        | Value |
|------------------|-------|
| Total models     | 523   |
| Mean score       | 54.6  |
| Median score     | 62.0  |
| Std. dev.        | 16.0  |
| Best score       | 74    |
| Models $\geq 60$ | 304   |
| Models $\geq 80$ | 0     |
| Max generation   | 11    |



(a) Distance to centroid.



(b) Pairwise distances.

Figure 2: Variation in generated proposals (price multiplier puzzle).

*Notes:* Panel (a) shows cosine distance between each proposal embedding and the average proposal embedding (using all-MiniLM-L6-v2 sentence embeddings); larger values indicate proposals that are farther from the center of the generated set. Panel (b) shows pairwise cosine distances across random proposal pairs; larger values indicate stronger dispersion in the explored idea space.

audit below makes concrete. The evolutionary search reaches generation 11. Model *fingerprints* (hashes of reported outputs and parameter counts) show that the 523 scored models embody 267 distinct output profiles; the top ten scores span five distinct fingerprints, i.e., the frontier contains a handful of genuinely different models, each rediscovered under several names.

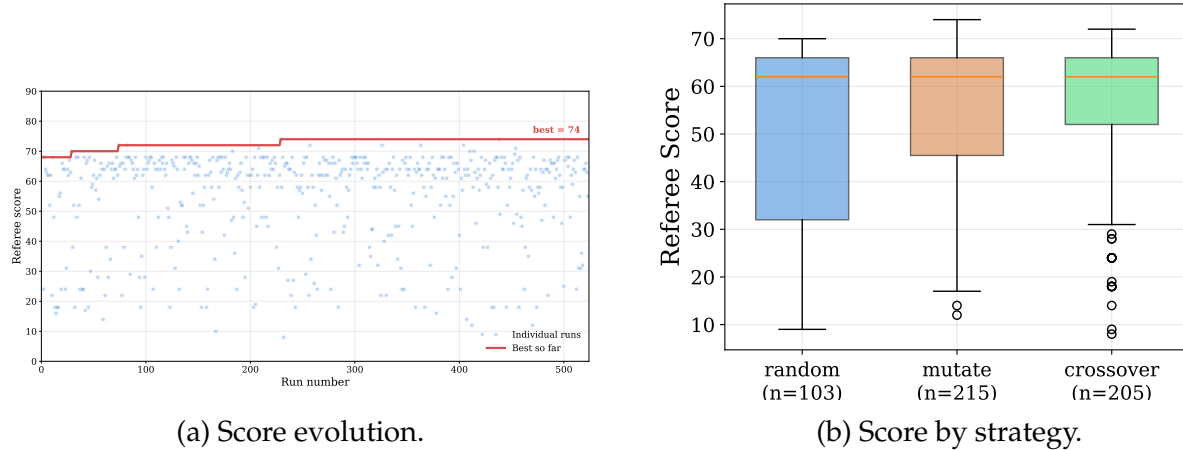


Figure 3: Score evolution across 523 scored runs (price multiplier puzzle).  
*Notes:* Panel (a) plots referee scores by run index, with the running maximum shown as a red line. Panel (b) reports score distributions by strategy using box plots. The figure visualizes the search dynamics over the full experiment horizon.

### 5.2.4 Heterogeneity of generated proposals

To assess whether the system explores diverse theoretical directions rather than producing minor variants of the same idea, we embed all 523 proposals using a sentence transformer (all-MiniLM-L6-v2) and compute cosine distances. Figure 2 shows the results. The mean cosine distance from each proposal to the centroid is 0.27, and the mean pairwise cosine distance is 0.46 (the absolute scale depends on the embedding model, but the pairwise distances indicate that proposals differ meaningfully in content rather than clustering around a single mechanism). The fingerprint counts above provide the complementary, output-based view: textual diversity is high, and even at the level of realized model outputs the run discovers hundreds of distinct profiles.

### 5.2.5 Score evolution

Figure 3 shows the score evolution over the experiment.

Table 3 reports strategy-level performance. Crossover achieves the highest mean referee score (56.3 over 205 models), mutate follows (54.3 over 215, and produces both best-

scoring models), and random generation trails (52.1 over 103)—though the random column is thinned by the proposal gate, which censors fresh proposals much more heavily than refinements of already-passed parents. Both top models are generation-4 and generation-5 mutations, so the frontier here was reached by evolutionary refinement rather than by a lucky fresh draw.

Table 3: Strategy Performance Comparison (Price Multiplier)

| Strategy  | $n$ | Mean | Median | Max |
|-----------|-----|------|--------|-----|
| Random    | 103 | 52.1 | 62.0   | 70  |
| Mutate    | 215 | 54.3 | 62.0   | 74  |
| Crossover | 205 | 56.3 | 62.0   | 72  |
| All       | 523 | 54.6 | 62.0   | 74  |

### 5.2.6 Lineage diversity under the current parent policy

Under the current score-squared policy, parents are drawn from the full population and crossover combines distinct lineages whenever possible. In the flagship run, the leading lineage accounts for 15% of scored descendants, 82% of parent selections come from outside it, every crossover combines two distinct lineages, and the top ten span five distinct fingerprints. These diagnostics show that selection pressure did not collapse the current search onto a single founding proposal.

### 5.2.7 Correlates of referee scores

To understand what drives the referee panel’s evaluation, we regress  $\log(1 + \text{referee score})$  on a set of observable features in four specifications (Table 4). Columns (1) and (2) exclude the quantitative composite score; columns (3) and (4) include it. Columns (2) and (4) add dummies for the evolutionary strategy. All specifications control for the length of the proposal text and the lengths of the referee’s written feedback fields, entered as  $\log(1 + \text{length})$ .

Table 4: Determinants of referee scores (Price Multiplier)

|                        | (1)                 | (2)                        | (3)                 | (4)                        |
|------------------------|---------------------|----------------------------|---------------------|----------------------------|
| Intercept              | 2.671***<br>(0.012) | 2.700***<br>(0.017)        | 1.960***<br>(0.026) | 1.978***<br>(0.027)        |
| Quantitative fit score |                     |                            | 0.016***<br>(0.001) | 0.016***<br>(0.001)        |
| Proposal length        | 1.851***<br>(0.008) | 1.872***<br>(0.012)        | 1.359***<br>(0.018) | 1.371***<br>(0.019)        |
| Strengths length       | 0.000<br>(0.000)    | -0.000000***<br>(0.000000) | 0.000<br>(0.000)    | 0.000000***<br>(0.000000)  |
| Weaknesses length      | 0.000<br>(0.000)    | 0.000<br>(0.000)           | 0.000<br>(0.000)    | -0.000000***<br>(0.000000) |
| Suggestions length     | 0.000<br>(0.000)    | 0.000<br>(0.000)           | 0.000<br>(0.000)    | 0.000<br>(0.000)           |
| Mutate (strategy)      |                     | -0.045<br>(0.036)          |                     | -0.035*<br>(0.018)         |
| Random (strategy)      |                     | -0.126**<br>(0.056)        |                     | -0.049**<br>(0.023)        |
| Strategy dummies       | No                  | Yes                        | No                  | Yes                        |
| N                      | 523                 | 523                        | 523                 | 523                        |
| $R^2$                  | -0.000              | 0.013                      | 0.791               | 0.794                      |

Notes: OLS regressions of  $\log(1 + \text{referee score})$  on text length features (log character counts of proposal, referee strengths, weaknesses, and suggestions), quantitative fit score, and (where indicated) strategy dummies. Heteroskedasticity-robust (HC1) standard errors are in parentheses. Significance levels are \*\*\*  $p < 0.01$ , \*\*  $p < 0.05$ , and \*  $p < 0.10$ .

The quantitative composite is by far the strongest predictor: the  $R^2$  rises from essentially zero in specifications (1)–(2) to about 0.79 in (3)–(4). Two features of this pattern are worth noting relative to a single-call referee. First, feedback lengths and strategy dummies carry almost no signal—the panel median appears to strip out the idiosyncratic verbosity effects that a single referee call exhibits. Second, the referee is aligned with, but not captured by, the mechanical composite: the audited models below show the panel systematically discounting high-fit models for specific, articulated reasons.

## 5.2.8 Top-Scoring Mechanisms

Table 5: Top 10 Discovered Models (Price Multiplier)

| Rank | Model                                                   | Strategy  | Gen. | Params | Fingerprint | Score |
|------|---------------------------------------------------------|-----------|------|--------|-------------|-------|
| 1    | Concave-Breadth Risk-and-Capacity Model                 | mutate    | 4    | 2      | 3a1992      | 74    |
| 2    | Nested Factor Risk with Inventory Costs (NFRIC)         | mutate    | 5    | 2      | fb8d5b      | 74    |
| 3    | Nested Risk-Congestion Capacity Model                   | crossover | 2    | 2      | fb8d5b      | 72    |
| 4    | Hierarchical Delegated-Capacity Model (HDCM)            | crossover | 3    | 2      | fb8d5b      | 72    |
| 5    | Shared-Capacity Ledger (SCL) Model                      | mutate    | 3    | 2      | b78790      | 72    |
| 6    | Hierarchical Effective-Capacity Inventory Model (HECIM) | mutate    | 6    | 2      | 965b87      | 72    |
| 7    | Inventory-Augmented Risk-Capacity (IARC) Model          | mutate    | 5    | 2      | fb8d5b      | 72    |
| 8    | Hierarchical Aggregate-Funding Capacity Model           | crossover | 8    | 3      | bc05f6      | 72    |
| 9    | Hierarchical Task-Load Intermediation Model (HTIM)      | mutate    | 4    | 2      | fb8d5b      | 71    |
| 10   | Inventory-Augmented Nested-Factor CARA (IANF-CARA)      | crossover | 6    | 2      | fb8d5b      | 71    |

*Notes:* Top 10 models by referee score from the price multiplier experiment. “Strategy” indicates whether the model was generated from scratch (random), by modifying a parent (mutate), or by combining two parents (crossover). “Gen.” is the model’s generation (lineage depth), “Params” its number of free parameters, and “Fingerprint” a hash of its reported outputs and parameter count — rows sharing a fingerprint are the same model rediscovered under a different name.

Table 5 lists the ten highest-scoring models. Two models tie at the top score of 74, and they are genuinely different mechanisms—one fingerprint each. We hand-audited both: every reported multiplier is the harness-computed comparative static of the model’s own declared multi-equation equilibrium system, both reproduce their reported outputs exactly at the stored parameters, and neither contains a smuggled target constant. Complete derivations of the two top-scoring mechanisms follow; both are short, because the verified derivation chain forces the mechanism into a small number of load-bearing equations.

### 1. Nested Factor Risk with Inventory Costs (NFRIC).

**Setup.** A representative competitive market maker with CARA–normal preferences (risk aversion  $\gamma$ ) absorbs exogenous, price-inelastic customer order flow  $q \in \mathbb{R}^N$  and

pays a per-stock quadratic inventory cost. Stocks are partitioned into granular groups of size  $N_g$  nested within coarse styles of size  $N_s$  (balanced hierarchy). Payoffs are  $X_i = \mu_i + f_{s(i)} + g_{g(i)} + \varepsilon_i$  with mutually independent style factors  $f_s \sim N(0, \sigma_{\text{style}}^2)$ , granular factors  $g_g \sim N(0, \sigma_{\text{style}}^2)$ , and idiosyncratic shocks  $\varepsilon_i \sim N(0, \sigma_{\text{idio}}^2)$ , so

$$\Sigma = \sigma_{\text{idio}}^2 I + \sigma_{\text{style}}^2 \sum_g \mathbf{1}_g \mathbf{1}_g' + \sigma_{\text{style}}^2 \sum_s \mathbf{1}_s \mathbf{1}_s'. \quad (10)$$

**Optimization and market clearing.** The intermediary solves

$$\max_h \left\{ (\mu - p)'h - \frac{\gamma}{2} h' \Sigma h - \frac{\theta}{2} h' h \right\}, \quad (11)$$

where  $\theta > 0$  is the inventory cost—a resource cost of maintaining positions (balance-sheet, financing, monitoring, shorting), not an extra preference parameter. The first-order condition gives demand  $h(p) = K^{-1}(\mu - p)$  with  $K = \gamma \Sigma + \theta I$  positive definite; market clearing  $h + q = \bar{h}$  then yields the equilibrium price  $p = \mu - K\bar{h} + Kq$  and the price-impact operator

$$\frac{\partial p}{\partial q} = K = \gamma \Sigma + \theta I. \quad (12)$$

Under the balanced hierarchy, the idiosyncratic, granular, and style subspaces are invariant under  $K$ , and the harness-verified multipliers are its eigenvalues on those subspaces:

$$M_{\text{idio}} = \gamma \sigma_{\text{idio}}^2 + \theta, \quad M_{6 \times 6} = \gamma \sigma_{\text{idio}}^2 + \gamma N_g \sigma_{\text{style}}^2 + \theta, \quad M_{\text{style}} = \gamma \sigma_{\text{idio}}^2 + \gamma (N_g + N_s) \sigma_{\text{style}}^2 + \theta. \quad (13)$$

**Why the gradient flattens.** The inventory cost  $\theta$  enters all three multipliers equally: it is a floor on price impact that diversification cannot remove, because it attaches to individual positions rather than to portfolio variance. As  $\sigma_{\text{idio}}^2 \rightarrow 0$  the idiosyncratic multiplier converges to  $\theta$ , not to zero. A common increase in  $\theta$  raises all multipliers one-for-one and compresses  $M_{\text{style}}/M_{\text{idio}}$  toward one—exactly the flattening the data demand.

**Over-identification.** Two parameters generate three multipliers, and equation (13) implies the parameter-free restriction

$$\frac{M_{\text{style}} - M_{6 \times 6}}{N_s} = \frac{M_{6 \times 6} - M_{\text{idio}}}{N_g} = \gamma \sigma_{\text{style}}^2. \quad (14)$$

In the data,  $(M_{\text{style}} - M_{6 \times 6})/N_s = 3.392/253 \approx 0.0134$  and  $(M_{6 \times 6} - M_{\text{idio}})/N_g = 0.890/63 \approx 0.0141$ : the restriction holds to within about 5%, which the model exploits rather than assumes.

**Calibration.** With  $\gamma \approx 1.258$  and  $\theta \approx 1.405$ , the model yields  $M_{\text{style}} \approx 5.944$ ,  $M_{6 \times 6} \approx 2.501$ ,  $M_{\text{idio}} \approx 1.643$ , and ratio  $\approx 3.62$  (fit 96.9, plausibility 100, parsimony 84). The referee panel (votes 72/74/78, median 74, “major revision”) accepts the equilibrium logic but docks the model for what it does not derive: *why* absorption capacity scales with group size is imposed through the factor structure rather than derived from deeper primitives, and the equal style/granular factor variances are a parsimony assumption. This is the frontier’s live open problem, not a hidden flaw—every model in the leading family shares it, and the panel prices it consistently at 72–78.

## 2. Concave-Breadth Risk-and-Capacity Model.

**Setup.** The same one-intermediary market-clearing structure, with two changes that *remove* free parameters. First, the capacity cost is tied structurally to idiosyncratic risk: the intermediary pays  $\frac{\gamma \sigma_{\text{idio}}^2}{2} h'h$ , eliminating the free cost scale. Second, shared-risk eigenvalues grow *concavely* with group breadth: factor loadings scale as  $n^{(\alpha-1)/2}$ , so the style and granular components of the covariance carry eigenvalues proportional to  $N_s^\alpha$  and  $N_g^\alpha$  with a single concavity parameter  $\alpha \in (0, 1)$ , rather than a separately fitted granular variance. The harness-verified multipliers are

$$M_{\text{idio}} = 2\gamma \sigma_{\text{idio}}^2, \quad M_{6 \times 6} = \gamma \sigma_{\text{style}}^2 N_g^\alpha + 2\gamma \sigma_{\text{idio}}^2, \quad M_{\text{style}} = \gamma \sigma_{\text{style}}^2 (N_g^\alpha + N_s^\alpha) + 2\gamma \sigma_{\text{idio}}^2. \quad (15)$$

**Mechanism.**  $\alpha$  measures how effectively shared risk aggregates with breadth:  $\alpha = 1$  recovers conventional linear factor scaling (and the steep benchmark gradient),  $\alpha = 0$  makes shared-risk pricing breadth-independent, and interior  $\alpha$  produces exactly the concave gradient in the data. Unlike a level-indexed capacity parameter, the aggregation gradient here is built from the *measured* volatilities and group sizes plus one fitted curvature.

**Calibration.** With  $\gamma \approx 4.184$  and  $\alpha \approx 0.764$ , the model yields  $M_{\text{style}} \approx 5.767$ ,  $M_{6 \times 6} \approx 2.658$ ,  $M_{\text{idio}} \approx 1.583$  (fit 83.3, plausibility 100, parsimony 84; panel votes 74/71/76, median 74). The fit is honest rather than exact—two parameters against three moments leave a genuine over-identifying tension of a few percent per moment—and the panel again withholds a higher score pending a primitive foundation for the concavity: what economic force makes effective risk-bearing breadth scale as  $n^{0.76}$ ?

**What the frontier looks like.** The two models are complementary answers to the same audited question. NFRIC explains the flattening with an additive inventory-cost *floor* and linear factor aggregation; the concave-breadth model explains it with sublinear *aggregation* and no separate floor. Both are genuine two-parameter competitive equilibria whose reported multipliers the harness derived, both reproduce exactly, and both are held at major revision for the same articulated reason: the aggregation structure that does the work is imposed, not derived from primitives. That convergent referee critique—consistent across models, panels, and proposer LLMs—is, to our knowledge, the sharpest automated statement of what a full resolution of this puzzle still requires.

### 5.3 Puzzle 2: Equity Term Structure (Dividend Strips)

The system’s second target is the equity term structure. This subsection reports the completed flagship experiment (randomized proposers, GPT-5.6-luna elsewhere, three-vote referee panel, score-squared parent selection; Table 1): 1,000 invocations. Every frontier

model has been reproduced from its stored artifacts and the deterministic dial detector (Section 4.3) has been applied to the entire top ten. None of the models below is advanced as a discovery.

### 5.3.1 The puzzle and calibration targets

**Definition of the puzzle.** Van Binsbergen et al. (2012) document that short-term equity claims (1–2 year dividend strips) have roughly twice the Sharpe ratio of the aggregate market, have large positive CAPM alphas, and have CAPM betas well below one. All leading consumption-based models (habit formation (Campbell and Cochrane, 1999), long-run risk (Bansal and Yaron, 2004), and rare disasters (Barro, 2006)) predict an upward-sloping term structure of equity risk premia (long-maturity claims earn higher returns), contradicting the data. The puzzle is therefore why short-term dividend claims are so well compensated relative to the market.

**Calibration targets.** The dividend strip puzzle requires matching ten moment conditions simultaneously (eight of them independent for fit purposes; the strip / market Sharpe ratio and CAPM alpha are algebraically implied by other outputs). The targets span unconditional moments (strip and market annual returns, monthly Sharpe ratios, CAPM alpha and beta, the risk-free rate) and conditional moments (the Sharpe ratio of strips relative to the market must be higher in recessions than expansions). Table 6 lists the targets and acceptable ranges. The harness owns the return derivations for this puzzle: strip and market return moments, and the risk-free rate, must be produced by the harness’s symbolic strip-return operator from the model’s own pricing kernel.

### 5.3.2 The experiment

Of the 1,000 invocations, 288 (29%) pass every gate. Attrition concentrates overwhelmingly in the fidelity gates: code review rejects 43% of all runs and the mathematical audit

Table 6: Dividend strip calibration targets

| Output                 | Description                      | Target | Range         |
|------------------------|----------------------------------|--------|---------------|
| strip_return_ann       | Strip annualized return          | 0.148  | [0.08, 0.22]  |
| market_return_ann      | Market annualized return         | 0.069  | [0.04, 0.10]  |
| strip_sharpe_monthly   | Strip monthly Sharpe             | 0.112  | [0.07, 0.18]  |
| market_sharpe_monthly  | Market monthly Sharpe            | 0.059  | [0.03, 0.10]  |
| sharpe_ratio           | SR(strip) / SR(market)           | 1.9    | [1.3, 2.5]    |
| capm_alpha_ann         | CAPM alpha (annualized)          | 0.091  | [0.04, 0.15]  |
| capm_beta              | CAPM beta of strip               | 0.50   | [0.3, 0.8]    |
| r_f_ann                | Real risk-free rate              | 0.006  | [0.001, 0.02] |
| sharpe_ratio_recession | Conditional SR ratio (recession) | 2.8    | [2.0, 4.0]    |
| sharpe_ratio_expansion | Conditional SR ratio (expansion) | 1.0    | [0.6, 1.4]    |

Notes: Unconditional targets from van Binsbergen et al. (2012). Conditional targets from Bansal et al. (2021) and Giglio et al. (2025). All returns and Sharpe ratios are in natural units (decimals, not percentages).

another 10% (Table 13)—a model must carry a ten-moment derivation faithfully from proposal through symbolic verification into code, and most failures are failures of that fidelity, not of ideas. (This puzzle’s reduced-form pricing-kernel formulation declares no required solution concept, so the proposal gate that dominates attrition on the other two puzzles does not run here; the filtering work shifts downstream.) The best referee score is 64, with eight further models at 60–62 (mean 35.7, median 35; Table 7); every referee panel holds the entire frontier at major revision. The 288 scored models embody 283 distinct fingerprints, and the top ten are ten distinct fingerprints—like the currency puzzle and unlike the price multiplier, the search has found no convergent family. Evolution is nonetheless doing real work: mutate and crossover models average 39.2 and 37.2 against 30.5 for fresh random draws (Table 8), and nine of the top ten models are refinements at generations one through six.

### 5.3.3 Current detector audit of conditional-Sharpe dials

The current deterministic detector identifies *private dials*—free parameters whose only independent effect on any fitted moment, verified by numerical perturbation at the optimum and after collapsing the puzzle’s declared output identities, is a single target. We

Table 7: Summary Statistics (Dividend Strips)

| Statistic        | Value |
|------------------|-------|
| Total models     | 288   |
| Mean score       | 35.7  |
| Median score     | 35.0  |
| Std. dev.        | 13.0  |
| Best score       | 64    |
| Models $\geq 60$ | 9     |
| Models $\geq 80$ | 0     |
| Max generation   | 8     |

Table 8: Strategy Performance Comparison (Dividend Strips)

| Strategy  | $n$ | Mean | Median | Max |
|-----------|-----|------|--------|-----|
| Random    | 100 | 30.5 | 28.0   | 59  |
| Mutate    | 118 | 39.2 | 38.5   | 62  |
| Crossover | 70  | 37.2 | 40.0   | 64  |
| All       | 288 | 35.7 | 35.0   | 64  |

apply it to the full top ten together with an exact reproduction check: all ten reproduce their stored outputs to machine precision from their saved code and parameters.

The scan cleanly splits the frontier, and the split runs straight through the top score. Five of the ten models fit the conditional Sharpe targets with a dial: the current best (64) moves *both* conditional Sharpe ratios through parameter directions that leave every other declared output numerically invariant, two models expose a recession-risk-premium parameter whose only independent effect is the recession Sharpe ratio, one does the same through a recession-variance parameter, and one through a joint direction in its regime-transition probabilities. In each case the remaining moments are genuinely produced by the model’s mechanism—the dial is a patch covering exactly the targets that require state dependence. The scan also flags inert parameters—free parameters that move no declared output at the optimum—in five of the ten models, dead degrees of freedom that a parameter count alone would misread as model complexity.

The clean current best is the *Aggregating Payout-Timing Model with Forward-Looking*

Table 9: Top 10 Discovered Models (Dividend Strips)

| Rank | Model                                                                 | Strategy  | Gen. | Params | Fingerprint | Score |
|------|-----------------------------------------------------------------------|-----------|------|--------|-------------|-------|
| 1    | Countercyclical Liquidity-Duration Model (CLD)                        | crossover | 5    | 10     | 5d60e4      | 64    |
| 2    | Timing-Neutral Market Liquidity Model                                 | mutate    | 1    | 7      | 0c9015      | 62    |
| 3    | The Aggregating Payout-Timing Model with Forward-Looking Risk Pricing | mutate    | 4    | 7      | 8bdc b1     | 62    |
| 4    | The State-Dependent Timing-Liquidity (SDTL) Model                     | crossover | 5    | 8      | 5d9761      | 62    |
| 5    | State-Normalized Timing-Liquidity (SNTL) Model                        | mutate    | 4    | 8      | 50efa4      | 61    |
| 6    | Regime-Switching Timing-Risk Model (RST-RM)                           | mutate    | 4    | 8      | f254b2      | 61    |
| 7    | Low-Beta Countercyclical Payout-Timing Model                          | mutate    | 1    | 6      | 481109      | 61    |
| 8    | Regime-Dependent Transitory Risk (RDTR) Model                         | crossover | 6    | 7      | 45f35a      | 61    |
| 9    | State-Dependent Timing-Exposure (SDTE) Model                          | mutate    | 5    | 8      | c3cd84      | 60    |
| 10   | The Systemic Solvency Stress (SSS) Model                              | random    | 0    | 8      | 962a26      | 59    |

*Notes:* Top 10 models by referee score from the dividend strips experiment. “Strategy” indicates whether the model was generated from scratch (random), by modifying a parent (mutate), or by combining two parents (crossover). “Gen.” is the model’s generation (lineage depth), “Params” its number of free parameters, and “Fingerprint” a hash of its reported outputs and parameter count — rows sharing a fingerprint are the same model rediscovered under a different name.

*Risk Pricing* (referee 62, a generation-4 mutation of a DeepSeek-proposed parent): the scan finds no dial and no inert parameter, its conditional Sharpe pattern emerges from a forward-looking timing-risk price interacting with regime persistence rather than from a regime-indexed knob, its quantitative composite (64.0) sits within two points of its referee score, and it posts an honest miss—its market return is 2.8% against a 6.9% target—rather than an exact fit. The contrast between the flagged 59–64 band and the clean 62 shows why current frontier rankings combine referee scores with deterministic integrity flags.

### 5.3.4 Why this puzzle is harder

The dividend strip puzzle is substantially harder for LLM-based theory search for three reasons. First, the number of fitted moments (8) far exceeds the price multiplier's (3). With that many independent constraints, a model needs a richer economic structure but must simultaneously maintain parsimony. Second, the conditional targets (recession vs. expansion Sharpe ratios) require state-dependent predictions, which most frameworks achieve through regime-switching or time-varying parameters, adding complexity—and, as the dial audit shows, those are exactly the targets the search learns to fake when it cannot produce them. Third, the derivation chain is longer: returns, Sharpe ratios, and CAPM coefficients must each be derived from the pricing kernel and carried through symbolic verification into code, and each additional derived moment is another chance to fail a fidelity gate—which is where over half of all runs die. The result is a parsimony trap: models that can match all the targets typically require many parameters, but models with few parameters cannot match the conditional patterns. Breaking out of the trap requires a parsimonious mechanism that generates conditional predictions from a small number of primitives—which the search, at 1,000 runs and one clean 62, has not found under full verification.

## 5.4 Puzzle 3: A Currency Premium Puzzle

The system’s third target is a currency premium puzzle formalized by Hassan et al. (2026). The flagship experiment is complete, but its top scores carry a documented caveat under the current integrity standard: the expected exchange-rate target is fitted by disguised free parameters. We report the experiment in full because it illustrates the system confronting a puzzle that is, by construction, close to unsolvable for a broad model class.

### 5.4.1 The puzzle and calibration targets

Let  $H$  denote the high-interest-rate currency and  $L$  the low;  $s$  is the log price of  $H$  in units of  $L$  (so  $\Delta s > 0$  means  $H$  appreciates), and the return to the carry trade “borrow  $L$ , lend  $H$ ” is  $(r_H - r_L) + \Delta s$ . Empirically (Hassan and Mano, 2019), essentially all of the carry-trade return comes from the interest differential: the average exchange-rate contribution is slightly *negative*. Hassan et al. (2026) show this is deeply puzzling for equilibrium models: within the class of models where bonds are priced by Euler equations with realistic risk compensation and an affine mean–variance link on the log pricing kernel, a persistent interest differential should be accompanied by a large expected exchange-rate movement—their baseline (a two-country long-run-risk economy) predicts an exchange-rate share of the carry return near 0.96, against roughly  $-0.4$  in the data.

Four independent moments are fitted; the carry return and its exchange-rate share are derived diagnostics, and Hansen–Jagannathan bounds for each country’s pricing kernel are reported to the referee unfitted (values far below  $\approx 0.5$  indicate unrealistically weak risk compensation). Table 10 lists the targets and acceptable ranges.

Deliberately, nothing beyond the data is imposed on the model class—no complete-markets or symmetric-preference requirement—because the paper’s no-go result covers the standard class, and hard-coding its conditions would define a puzzle with (nearly) no solution. The proposal gate requires only that interest rates *and the exchange rate* be pinned down by the stated equilibrium; the harness owns the pricing-kernel moments per

Table 10: Currency premium calibration targets

| Output                 | Description                                 | Target  | Range (95% CI)    | Fitted? |
|------------------------|---------------------------------------------|---------|-------------------|---------|
| interest_differential  | $r_H - r_L$                                 | 0.0476  | [0.0130, 0.0846]  | yes     |
| fx_change              | $E[\Delta s]$                               | -0.0130 | [-0.0382, 0.0060] | yes     |
| currency_return        | carry return (derived)                      | 0.0346  | [0.0118, 0.0554]  | no      |
| fx_share               | $E[\Delta s]/\text{carry return (derived)}$ | -0.37   | [-1.16, 0.23]     | no      |
| r_f_avg                | average risk-free rate                      | 0.01    | [-0.01, 0.03]     | yes     |
| r_f_vol                | risk-free rate volatility                   | 0.01    | [0, 0.03]         | yes     |
| hj_bound_H, hj_bound_L | $\sigma(M)/E[M]$ per kernel (diagnostic)    | —       | —                 | no      |

Notes: Empirical moments from Hassan and Mano (2019) (16 currencies, 1983–2010, annual), as targeted by Hassan et al. (2026).  $H$  denotes the high-interest-rate currency,  $L$  the low;  $s$  is the log price of  $H$  in units of  $L$ . Calibration inputs (fixed, never optimized): world consumption growth mean  $\mu_c = 0.018$  and volatility  $\sigma_c = 0.027$  (annual); every cross-country difference is a model parameter. Hansen–Jagannathan bounds are reported to the referee but not fitted.

country (risk-free rates, kernel variances, and Hansen–Jagannathan bounds must come from the symbolic *sdf-moments* operator applied to each country’s kernel).

#### 5.4.2 The experiment and what the audit found

The experiment uses the same flagship recipe as the price multiplier (randomized proposers, GPT-5.6-luna elsewhere, three-vote referee panel, score-squared parent selection), run in two batches of 500: 1,000 invocations, of which 409 (41%) pass every gate and enter the registry. The proposal gate is the dominant filter, rejecting 43% of all runs (Table 13)—on this puzzle the gate’s rationales are substantively right, in each case naming a specific exchange-rate block that the proposal asserts rather than determines. The best referee score is 68, reached by a single model, with 22 models at 62 (mean 39.7; Table 11); every panel holds the top models at major revision or below. Strikingly, all 409 scored models carry distinct fingerprints—unlike the price multiplier, the search has found no convergent family on this puzzle.

We hand-audited six frontier models: five tied at 62 and the best at 68. On the harness-owned side every audit is clean: every rate-side moment is genuinely Euler-derived through the symbolic operator, per country, and every audited model reproduces its reported outputs exactly. The caveat concerns the one fitted target the harness does *not*

Table 11: Summary Statistics (Currency Premium)

| Statistic        | Value |
|------------------|-------|
| Total models     | 409   |
| Mean score       | 39.7  |
| Median score     | 42.0  |
| Std. dev.        | 15.8  |
| Best score       | 68    |
| Models $\geq 60$ | 34    |
| Models $\geq 80$ | 0     |
| Max generation   | 10    |

own: the expected exchange-rate change. In three of the five audited 62s, that target is fitted by a free parameter whose only effect on any fitted moment—verified by numerical perturbation at the optimum—is to move the exchange-rate change itself: a private dial disguised as economic structure (an exogenous revaluation drift, a two-parameter conversion wedge, a three-parameter settlement friction). The referee panels saw the pattern in every case, said so in their reports, and capped those models at or below 62.

The 68 is the same dial, better disguised. The model fits the exchange-rate change with a “permanent institutional mean gap” parameter  $d$  whose optimized value equals the target to sixteen digits ( $\text{fx\_change} = -d$ ), and which *appears* to move four fitted moments—but the extra appearances are pure pass-throughs of the puzzle’s own accounting identity (the carry return is *defined* as the interest differential plus the exchange-rate change): perturbing  $d$  leaves the carry return unchanged to machine precision, so net of the identity its only independent effect is the dialed target. All three referee votes identified the cancellation explicitly (one scored it 58/reject), yet the median landed at 68. Under the current design, the identity-aware private-dial detector makes this integrity classification deterministic rather than leaving its weight to referee judgment. The current reading is therefore that the 58–68 band solves the interest-rate side genuinely and fits the exchange-rate side by assumption.

Table 12: Cross-puzzle comparison

|                               | Price Multiplier<br>(flagship) | Dividend Strips<br>(flagship)     | Currency Premium<br>(flagship)    |
|-------------------------------|--------------------------------|-----------------------------------|-----------------------------------|
| Fitted moments                | 3                              | 8 (10 outputs)                    | 4                                 |
| Conditional targets           | No                             | Yes                               | No                                |
| Harness-owned definition      | multipliers                    | return moments                    | kernel moments                    |
| Pipeline invocations          | 1,000                          | 1,000                             | 1,000                             |
| Scored models                 | 523                            | 288                               | 409                               |
| Completion rate               | 52%                            | 29%                               | 41%                               |
| Largest filter                | proposal gate (21%)            | code review (43%)                 | proposal gate (43%)               |
| Mean referee score            | 54.6                           | 35.7                              | 39.7                              |
| Best referee score            | 74                             | 64 (dialed; clean 62)             | 68 (dialed; see text)             |
| Distinct fingerprints, top 10 | 5                              | 10                                | 10                                |
| Top models audited            | clean                          | 5 of 10 dial a conditional target | rate side clean; fx target dialed |
| Referee                       | luna panel (3)                 | luna panel (3)                    | luna panel (3)                    |

*Notes:* Comparison across the three puzzles; all three experiments use the flagship recipe of Table 1, and every referee score is the median of a three-vote GPT-5.6-luna panel. “Fitted moments” counts the independent targets the optimizer fits. “Harness-owned definition” names the quantities that must be produced by the harness’s symbolic operators rather than derived by the model’s own code. “Largest filter” is the pipeline stage rejecting the most runs (the dividend-strip variant declares no required solution concept, so the proposal gate does not run there).

Table 13: Pipeline attrition by stage

| Stage                              | Price Multiplier |     | Dividend Strips |     | Currency Premium |     |
|------------------------------------|------------------|-----|-----------------|-----|------------------|-----|
|                                    | <i>n</i>         | %   | <i>n</i>        | %   | <i>n</i>         | %   |
| Total invocations                  | 1000             | 100 | 1000            | 100 | 1000             | 100 |
| Failed — problem analysis          | 0                | 0   | 0               | 0   | 1                | 0   |
| Failed — proposal gate             | 213              | 21  | 0               | 0   | 429              | 43  |
| Failed — SymPy derivation          | 37               | 4   | 34              | 3   | 13               | 1   |
| Failed — transcription gate        | 76               | 8   | 26              | 3   | 15               | 2   |
| Failed — math audit                | 1                | 0   | 99              | 10  | 11               | 1   |
| Failed — code generation           | 35               | 4   | 33              | 3   | 36               | 4   |
| Failed — code review               | 73               | 7   | 429             | 43  | 58               | 6   |
| Failed — optimizer                 | 21               | 2   | 40              | 4   | 12               | 1   |
| Failed — code-SymPy reconciliation | 13               | 1   | 24              | 2   | 0                | 0   |
| Failed — timeout                   | 7                | 1   | 15              | 2   | 12               | 1   |
| Failed — unknown                   | 1                | 0   | 12              | 1   | 4                | 0   |
| Scored (in registry)               | 523              | 52  | 288             | 29  | 409              | 41  |

*Notes:* Pipeline attrition per puzzle, from each track’s experiment log. “Total invocations” is the number of pipeline runs attempted; each failed row counts runs rejected at that stage or gate; “Scored” is the number that passed every gate and entered the model registry. Rejection — not just retry — at the proposal gate, transcription gate, code review, constraint gate, and code-SymPy reconciliation is deliberate: models that cannot demonstrate a faithful derivation-to-code chain are discarded rather than scored.

## 5.5 Cross-Puzzle Comparison

Table 12 compares the three experiments and Table 13 reports full pipeline attrition. Three patterns organize the comparison. First, *where each puzzle binds is diagnostic*: the price multiplier’s and currency puzzle’s filter is the proposal gate (fresh ideas are plentiful but often assert the result; on the currency puzzle the exchange rate is the hard half of any proposal, and the gate correctly demands it be determined in equilibrium), while the dividend strip’s filters are the fidelity gates (ideas survive, but ten derived moments rarely make it through code review and the mathematical audit intact). Second, *evolution refines on every puzzle where the parent pool clears a floor*: the price multiplier’s best models are generation-4 and generation-5 mutations, and on the dividend strip—despite a much weaker pool—mutate and crossover outperform fresh draws by about eight points on average and supply nine of the top ten models, including the clean best at generation four. Third, *the highest raw score is not the frontier*: on both caveated puzzles the top raw score belongs to a model the dial audit flags, and the defensible best sits below it—which is precisely why the pipeline’s deterministic scans, not its scores, define what the system has found.

### 5.5.1 Economic Validity and Novelty

Multiple mechanisms can fit the same empirical targets, so the discriminating evidence is the models’ *additional* implications. For the price multiplier, both top mechanisms are directly testable on flow data: the NFRIC over-identifying restriction (14) is a parameter-free prediction relating the multiplier gradient to group sizes, and the concave-breadth model predicts a specific curvature ( $n^{0.76}$ ) in how price impact aggregates with breadth. On novelty, the referee panel treats the inventory-cost floor and the concave-breadth aggregation as distinct contributions and, in both cases, points its suggestions at the same open problem: deriving the aggregation structure from primitives rather than imposing it.

## 6 Discussion

### 6.1 What LLMs Can and Cannot Do

Our experiments across three puzzles reveal both capabilities and limitations of LLM-based theory discovery.

On the capability side, LLMs generate varied economic mechanisms, including ambiguity aversion, capacity constraints, crowding effects, information risk, and regime-switching liquidity, without explicit prompting for specific ideas. Models are expressed as precise mathematical formulas that can be implemented and tested. The verification stages catch a substantial fraction of derivation errors before they propagate into code. The simulated referee identifies weaknesses such as lack of micro-foundations and need for robustness checks. The persona system generates qualitatively different theoretical perspectives: a “physicist” persona proposes conservation-law models, a “behavioral” persona explores bounded rationality, and a “minimalist” persona seeks single-equation explanations.

The central design lesson is that numerical fit alone does not establish that a model computes the object a target defines. The current integrity chain therefore gives the harness ownership of target definitions, derives headline quantities with symbolic operators, reconciles verified formulas against scored code, rejects hard-coded targets and tunable state realizations, and scans for free parameters whose only independent effect is a single fitted target. The currency and dividend-strip results show why the final scan matters: their highest raw scores concentrate on the least mechanically constrained targets—the exchange-rate change in one case and conditional Sharpe ratios in the other. Under the current standard these models remain informative about the genuinely derived portions of each puzzle, but are not classified as solutions.

On the theoretical side, LLMs face the same fundamental challenge as human theorists: multiple mechanisms can produce the same empirical targets, and fitting data does

not establish that a mechanism is correct. Small modifications to good models often break their calibration. Despite prompts encouraging novel frameworks, the majority of price-multiplier models remain CARA-intermediary variants. The model *fingerprint* machinery makes the flip side measurable: of 523 scored price-multiplier models, 267 have distinct output profiles, but the audited frontier collapses to two genuinely different ideas, each rediscovered under many names. A search that reports “hundreds of models” without output-level deduplication overstates its harvest.

The dividend strip experiment reveals a subtler limitation: when a puzzle contains targets the surviving mechanisms cannot genuinely produce, evolution can refine *toward the dial*. Mutation and crossover outperform fresh draws there by about eight referee points on average and supply nine of the top ten models, but five of those ten, including the raw best, fit the conditional Sharpe targets with a private dial. The price-multiplier experiment shows the benign regime: with a strong parent pool and every target harness-owned, both top models are genuine generation-4 and generation-5 mutations. Under the current score-squared policy, the leading lineage accounts for only 15% of scored descendants and the top ten span five distinct fingerprints, so the frontier remains diverse while refinement concentrates search on stronger mechanisms.

The LLM may be reproducing known solutions from its training data. The proposers’ training corpora almost certainly include the asset pricing literature, including the papers that document our target puzzles. However, these puzzles are *open*: no published resolution exists for the price-multiplier scaling gap, the dividend strip term structure, or the currency premium puzzle. The LLMs have read the building blocks (CARA and CRRA preferences, inventory costs, Bayesian learning, portfolio constraints, regime-switching models) but not the specific combinations proposed here, because those combinations had not been published. This recombination is the same creative process by which human theorists work: Bansal and Yaron (2004) combined long-run consumption risk with Epstein–Zin preferences, neither of which was new, to resolve the equity premium puz-

zle. As Fleming (2001) argues, novel recombination of existing components is the primary mode of scientific and technological creativity.

A further consideration concerns the evaluation design. The referee receives the quantitative composite as part of its input, and the correlation between the two is high ( $R^2 \approx 0.79$  for the price-multiplier flagship): the referee’s level is largely anchored on the mechanical composite. Three features discipline the residual judgment. First, the primary quality signal—fit, half the total weight—comes from deterministic code checking model outputs against empirical moments, which no LLM bias can influence; under the integrity chain, the outputs being checked are themselves verified. Second, the recorded score is the median of three independent referee calls, which strips out single-call verbosity effects (feedback lengths, which predicted scores under a single-call referee, carry essentially no signal under the panel). Third, and most important, the audited cases show the referee subtracting from the composite for articulated reasons: the top price-multiplier models carry composites of 88–95 against referee scores of 72–78, with the gap explained in the reports (imposed aggregation structure, unmicrofounded costs), and in the currency experiment every panel identified the exchange-rate dial in every audited top model—though the score they attached to the same finding varied from 58 to 68 across votes, which is exactly why such checks must be deterministic and shown to every judge rather than left to referee weighting. Referee generosity does differ systematically across backends, so we never compare scores across referee models. The referee’s other contribution is qualitative: its feedback is passed to the mutate operator so that subsequent proposals address specific shortcomings.

Finally, our results depend on design choices whose sensitivity we have only partly explored. The flagship experiments randomize the proposer across five LLM families—frontier scores are strikingly flat across proposers, though the proposal gate censors them unevenly—and report lineage and fingerprint diagnostics under the current score-squared parent policy. Unexplored sensitivities remain: the scoring weights (50/25/25 for fit/-

plausibility/parsimony), the persona set, prompt wording, and the referee backend, which shifts score *levels* even though it appears not to reorder audited models.

## 6.2 Comparison to Human Theorists

The economics of LLM-based discovery differ fundamentally from human research. The current system parallelizes complete gated runs across as many as 128 workers, allowing hundreds of candidate mechanisms to be generated, derived, coded, calibrated, and evaluated in a single experiment, while human cognition is largely sequential.

The system automates the creative generation phase of theoretical research, from puzzle to scored candidate mechanisms, without human intervention. The remaining human role is downstream: selecting promising theories for development, exploring their empirical implications, refining mechanisms with domain expertise, and auditing the current pipeline and its frontier. Automating generation shifts human comparative advantage toward curation, development, and adversarial verification of the search process.

## 6.3 Cross-Puzzle Lessons

The contrast across the three experiments is instructive. The price multiplier puzzle, with three targets, a clear benchmark failure, and a harness-ownable definition of its target quantity, is well-suited to automated search: two-parameter models can match all targets, the evolutionary loop activates early, and the system converges on a small number of distinct mechanisms. The dividend strip puzzle, with eight fitted moments including conditional patterns, pushes models toward the parsimony boundary and its long derivation chain concentrates failures in the fidelity gates. The currency puzzle is hard in a third way: the model class that could solve it is nearly empty by construction, and the search’s response was to attack the one target whose definition is not yet harness-owned.

Puzzles are most tractable for LLM-based search when they (i) have a small number

of tightly specified targets, (ii) admit parsimonious solutions in principle, (iii) allow early runs to produce “good enough” models that seed evolutionary refinement, and (iv) have target quantities whose definitions can be executed by the harness rather than trusted to the model. Condition (iv) is the one this project adds to the list, and the currency and dividend-strip experiments show what happens at its boundary: the search fits the least-constrained target by assumption, with a disguise good enough that only a deterministic scan separates it from structure.

## 6.4 Implications for Economic Research

Our findings suggest several implications for the practice of economic research. The time from puzzle identification to a set of verified candidate mechanisms could shrink from years to days. Robust evaluation pipelines become essential—and our results sharpen what “robust” means: not only parsimony pressure against over-parameterized curve-fitting, but executable definitions of the target quantities and deterministic provenance checks from derivation to reported number. Without them, the search optimizes the appearance of success; with them, the surviving models are few, small, and honest about what they have not yet derived—and the referee’s convergent critique of those models reads as a research agenda.

## 7 Conclusion

LLMs can discover economic theories that explain empirical puzzles—provided the search is verified end to end. Applied to the price-multiplier puzzle under the current integrity chain, the system discovers two genuinely distinct two-parameter equilibrium mechanisms (an inventory-cost floor on price impact and concave risk aggregation across breadth) that match all three empirical targets, satisfy an over-identifying restriction in the published estimates, and survive line-by-line hand audits. Experiments on a currency pre-

mium puzzle and the equity term structure (both complete at 1,000 invocations) show the system confronting harder targets: under the current integrity standard neither is yet solved, because their highest raw scores rely on free parameters serving as private dials on the least-constrained targets. The current deterministic sensitivity scan identifies these models explicitly.

Four findings emerge. First, LLMs can propose fundamentally different mechanisms rather than merely tweaking existing models, but output-level fingerprinting shows that an apparent multitude of models typically collapses to a smaller set of distinct ideas. Second, verification is the substance of the method: harness-owned definitions, symbolic comparative statics, code-derivation reconciliation, AST checks, and output-sensitivity scans ensure that quantitative fit comes from the stated mechanism. Third, evolutionary refinement genuinely improves scores, but it optimizes whatever the score rewards; the current score-squared, cross-lineage parent policy preserves frontier diversity, while deterministic integrity flags distinguish structural improvement from private calibration dials. Fourth, the referee panel’s convergent critique of the surviving models—that the aggregation structure doing the work is imposed rather than derived from primitives—functions as an automatically generated statement of what a full resolution of the puzzle still requires.

The pipeline, not any single LLM, does the heavy lifting: proposal screening, symbolic derivation with harness-owned definitions, mathematical auditing, reviewed translation to code, deterministic reconciliation, and panel-based peer review filter LLM proposals into calibrated, audited economic theories.

The approach applies to any empirical puzzle with well-defined quantitative targets—most effectively, we find, when the definitions of those targets can themselves be executed by the harness. As LLMs improve and evaluation pipelines mature, automated theory search can expand the set of candidate explanations that receive serious scrutiny.

## References

- Abaluck, Jason, Robert Pless, Nirmal Ravi, Anja Sautmann, and Aaron Schwartz (Jan. 2026). *Does LLM Assistance Improve Healthcare Delivery? An Evaluation Using On-site Physicians and Laboratory Tests*. DOI: [10.3386/w34660](https://doi.org/10.3386/w34660). National Bureau of Economic Research: [34660](https://www.nber.org/papers/w34660). URL: <https://www.nber.org/papers/w34660> (visited on 03/13/2026). Pre-published.
- Amabile, Teresa M. (1983). "The Social Psychology of Creativity: A Componential Conceptualization". In: *Journal of Personality and Social Psychology* 45.2, pp. 357–376. ISSN: 1939-1315. DOI: [10.1037/0022-3514.45.2.357](https://doi.org/10.1037/0022-3514.45.2.357).
- Asirvatham, Hemanth, Elliott Moksiki, and Andrei Shleifer (Feb. 2026). *GPT as a Measurement Tool*. DOI: [10.3386/w34834](https://doi.org/10.3386/w34834). National Bureau of Economic Research: [34834](https://www.nber.org/papers/w34834). URL: <https://www.nber.org/papers/w34834> (visited on 03/13/2026). Pre-published.
- Bansal, Ravi, Shane Miller, Dongho Song, and Amir Yaron (Dec. 1, 2021). "The Term Structure of Equity Risk Premia". In: *Journal of Financial Economics* 142.3, pp. 1209–1228. ISSN: 0304-405X. DOI: [10.1016/j.jfineco.2021.05.043](https://doi.org/10.1016/j.jfineco.2021.05.043). URL: <https://www.sciencedirect.com/science/article/pii/S0304405X21002361> (visited on 03/13/2026).
- Bansal, Ravi and Amir Yaron (2004). "Risks for the Long Run: A Potential Resolution of Asset Pricing Puzzles". In: *The Journal of Finance* 59.4, pp. 1481–1509. ISSN: 1540-6261. DOI: [10.1111/j.1540-6261.2004.00670.x](https://doi.org/10.1111/j.1540-6261.2004.00670.x). URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.2004.00670.x> (visited on 03/13/2026).
- Barro, Robert J. (Aug. 1, 2006). "Rare Disasters and Asset Markets in the Twentieth Century\*". In: *The Quarterly Journal of Economics* 121.3, pp. 823–866. ISSN: 0033-5533. DOI: [10.1162/qjec.121.3.823](https://doi.org/10.1162/qjec.121.3.823). URL: <https://doi.org/10.1162/qjec.121.3.823> (visited on 03/13/2026).
- Bini, Pietro, Lin William Cong, Xing Huang, and Lawrence J. Jin (Jan. 2026). *Behavioral Economics of AI: LLM Biases and Corrections*. DOI: [10.3386/w34745](https://doi.org/10.3386/w34745). National Bureau of

- Economic Research: 34745. URL: <https://www.nber.org/papers/w34745> (visited on 03/13/2026). Pre-published.
- Boiko, Daniil A., Robert MacKnight, Ben Kline, and Gabe Gomes (Dec. 2023). “Autonomous Chemical Research with Large Language Models”. In: *Nature* 624.7992, pp. 570–578. ISSN: 1476-4687. DOI: [10.1038/s41586-023-06792-0](https://doi.org/10.1038/s41586-023-06792-0). URL: <https://www.nature.com/articles/s41586-023-06792-0> (visited on 03/13/2026).
- Campbell, Donald T. (1960). “Blind Variation and Selective Retentions in Creative Thought as in Other Knowledge Processes”. In: *Psychological Review* 67.6, pp. 380–400. ISSN: 1939-1471. DOI: [10.1037/h0040373](https://doi.org/10.1037/h0040373).
- Campbell, John Y. and John H. Cochrane (Apr. 1999). “By Force of Habit: A Consumption-Based Explanation of Aggregate Stock Market Behavior”. In: *Journal of Political Economy* 107.2, pp. 205–251. ISSN: 0022-3808. DOI: [10.1086/250059](https://doi.org/10.1086/250059). URL: <https://www.journals.uchicago.edu/doi/10.1086/250059> (visited on 03/13/2026).
- Chatterji, Aaron, Thomas Cunningham, David J. Deming, Zoe Hitzig, Christopher Ong, Carl Yan Shan, and Kevin Wadman (Sept. 2025). *How People Use ChatGPT*. DOI: [10.3386/w34255](https://doi.org/10.3386/w34255). National Bureau of Economic Research: 34255. URL: <https://www.nber.org/papers/w34255> (visited on 03/13/2026). Pre-published.
- Fleming, Lee (Jan. 2001). “Recombinant Uncertainty in Technological Search”. In: *Management Science* 47.1, pp. 117–132. ISSN: 0025-1909. DOI: [10.1287/mnsc.47.1.117.10671](https://doi.org/10.1287/mnsc.47.1.117.10671). URL: <https://pubsonline.informs.org/doi/10.1287/mnsc.47.1.117.10671> (visited on 03/13/2026).
- Fortunato, Santo, Carl T. Bergstrom, Katy Börner, James A. Evans, Dirk Helbing, Staša Milojević, Alexander M. Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, Alessandro Vespignani, Ludo Waltman, Dashun Wang, and Albert-László Barabási (Mar. 2, 2018). “Science of Science”. In: *Science* 359.6379, eaao0185. DOI: [10.1126/science.aao0185](https://doi.org/10.1126/science.aao0185). URL: <https://www.science.org/doi/10.1126/science.aao0185> (visited on 03/13/2026).

- Foster, Jacob G., Andrey Rzhetsky, and James A. Evans (Oct. 1, 2015). “Tradition and Innovation in Scientists’ Research Strategies”. In: *American Sociological Review* 80.5, pp. 875–908. ISSN: 0003-1224. DOI: [10.1177/0003122415601618](https://doi.org/10.1177/0003122415601618). URL: <https://doi.org/10.1177/0003122415601618> (visited on 03/13/2026).
- Gabaix, Xavier and Ralph S. J. Koijen (Dec. 23, 2023). *In Search of the Origins of Financial Fluctuations: The Inelastic Markets Hypothesis*. DOI: [10.2139/ssrn.3686935](https://ssrn.com/abstract=3686935). Social Science Research Network: [3686935](https://ssrn.com/abstract=3686935). URL: <https://papers.ssrn.com/abstract=3686935> (visited on 03/13/2026). Pre-published.
- Galiani, Sebastian, Ramiro H. Gálvez, Franco Mettola La Giglia, and Raul A. Sosa (Jan. 2026). *Measuring Efficiency and Equity Framing in Economics Research: LLM-Based Evidence from 1950 to 2021*. DOI: [10.3386/w34714](https://www.nber.org/papers/w34714). National Bureau of Economic Research: [34714](https://www.nber.org/papers/w34714). URL: <https://www.nber.org/papers/w34714> (visited on 03/13/2026). Pre-published.
- Giglio, Stefano, Dacheng Xiu, and Dake Zhang (2025). “Test Assets and Weak Factors”. In: *The Journal of Finance* 80.1, pp. 259–319. ISSN: 1540-6261. DOI: [10.1111/jofi.13415](https://onlinelibrary.wiley.com/doi/abs/10.1111/jofi.13415). URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/jofi.13415> (visited on 03/13/2026).
- Glosten, Lawrence R. and Paul R. Milgrom (Mar. 1, 1985). “Bid, Ask and Transaction Prices in a Specialist Market with Heterogeneously Informed Traders”. In: *Journal of Financial Economics* 14.1, pp. 71–100. ISSN: 0304-405X. DOI: [10.1016/0304-405X\(85\)90044-3](https://www.sciencedirect.com/science/article/pii/0304405X85900443). URL: <https://www.sciencedirect.com/science/article/pii/0304405X85900443> (visited on 03/13/2026).
- Goetzmann, William N., Dasol Kim, and Robert J. Shiller (June 2024). *Emotions and Subjective Crash Beliefs*. DOI: [10.3386/w32589](https://www.nber.org/papers/w32589). National Bureau of Economic Research: [32589](https://www.nber.org/papers/w32589). URL: <https://www.nber.org/papers/w32589> (visited on 03/13/2026). Pre-published.
- Hansen, Stephen, Peter John Lambert, Nicholas Bloom, Steven J. Davis, Raffaella Sadun, and Bledi Taska (Mar. 2023). *Remote Work across Jobs, Companies, and Space*. DOI: [10.3386/w30000](https://www.nber.org/papers/w30000).

- 3386/w31007. National Bureau of Economic Research: 31007. URL: <https://www.nber.org/papers/w31007> (visited on 03/13/2026). Pre-published.
- Hasbrouck, Joel (2009). “Trading Costs and Returns for U.S. Equities: Estimating Effective Costs from Daily Data”. In: *The Journal of Finance* 64.3, pp. 1445–1477. ISSN: 1540-6261. DOI: [10.1111/j.1540-6261.2009.01469.x](https://doi.org/10.1111/j.1540-6261.2009.01469.x). URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.2009.01469.x> (visited on 03/13/2026).
- Hassan, Tarek A. and Rui C. Mano (2019). “Forward and Spot Exchange Rates in a Multi-Currency World”. In: *The Quarterly Journal of Economics* 134.1, pp. 397–450.
- Hassan, Tarek A., Thomas M. Mertens, and Jingye Wang (Aug. 2026). *A Currency Premium Puzzle*. Working Paper 35572. National Bureau of Economic Research.
- Heckman, James J. and Burton Singer (May 2017). “Abducting Economics”. In: *American Economic Review* 107.5, pp. 298–302. ISSN: 0002-8282. DOI: [10.1257/aer.p20171118](https://doi.org/10.1257/aer.p20171118). URL: <https://www.aeaweb.org/articles?id=10.1257/aer.p20171118> (visited on 03/13/2026).
- Horton, John J. (2023). “Large Language Models as Simulated Economic Agents: What Can We Learn from Homo Silicus?” In: *SSRN Electronic Journal*. ISSN: 1556-5068. DOI: [10.2139/ssrn.4413859](https://doi.org/10.2139/ssrn.4413859). URL: <http://dx.doi.org/10.2139/ssrn.4413859>.
- Hubert, Thomas, Rishi Mehta, Laurent Sartran, Miklós Z. Horváth, Goran Žužić, Eric Wieser, Aja Huang, Julian Schrittwieser, Yannick Schroecker, Hussain Masoom, Otavia Bertolli, Tom Zahavy, Amol Mandhane, Jessica Yung, Iuliya Beloshapka, Borja Ibarz, Vivek Veeriah, Lei Yu, Oliver Nash, Paul Lezeau, Salvatore Mercuri, Calle Sönne, Bhavik Mehta, Alex Davies, Daniel Zheng, Fabian Pedregosa, Yin Li, Ingrid von Glehn, Mark Rowland, Samuel Albanie, Ameya Velingker, Simon Schmitt, Edward Lockhart, Edward Hughes, Henryk Michalewski, Nicolas Sonnerat, Demis Hassabis, Pushmeet Kohli, and David Silver (Nov. 12, 2025). “Olympiad-Level Formal Mathematical Reasoning with Reinforcement Learning”. In: *Nature*, pp. 1–3. ISSN: 1476-4687. DOI: [10.1038/s41586-025-05000-0](https://doi.org/10.1038/s41586-025-05000-0).

1038/s41586-025-09833-y. URL: <https://www.nature.com/articles/s41586-025-09833-y> (visited on 03/13/2026).

Jabarian, Brian and Alex Imas (Sept. 2025). *Artificial Writing and Automated Detection*. DOI: 10.3386/w34223. National Bureau of Economic Research: 34223. URL: <https://www.nber.org/papers/w34223> (visited on 03/13/2026). Pre-published.

Jumper, John, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Židek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis (Aug. 2021). “Highly Accurate Protein Structure Prediction with AlphaFold”. In: *Nature* 596.7873, pp. 583–589. ISSN: 1476-4687. DOI: 10.1038/s41586-021-03819-2. URL: <https://www.nature.com/articles/s41586-021-03819-2> (visited on 03/13/2026).

Korinek, Anton (Feb. 2023). *Language Models and Cognitive Automation for Economic Research*. DOI: 10.3386/w30957. National Bureau of Economic Research: 30957. URL: <https://www.nber.org/papers/w30957> (visited on 03/13/2026). Pre-published.

— (Nov. 2024). *Generative AI for Economic Research: LLMs Learn to Collaborate and Reason*. DOI: 10.3386/w33198. National Bureau of Economic Research: 33198. URL: <https://www.nber.org/papers/w33198> (visited on 12/26/2024). Pre-published.

— (Sept. 2025). *AI Agents for Economic Research*. DOI: 10.3386/w34202. National Bureau of Economic Research: 34202. URL: <https://www.nber.org/papers/w34202> (visited on 03/13/2026). Pre-published.

Kyle, Albert S. (Nov. 1985). “Continuous Auctions and Insider Trading”. In: *Econometrica* 53.6, p. 1315. ISSN: 00129682. DOI: 10.2307/1913210.

- Li, Jiacui and Zihan Lin (2026). “Price Multipliers Are Larger for Less Diversifiable Order Flows”. In: *The Journal of Finance*. URL: <https://papers.ssrn.com/abstract=4038664> (visited on 03/13/2026). Forthcoming.
- Lopez-Lira, Alejandro (Apr. 2025). *Can Large Language Models Trade? Testing Financial Theories with LLM Agents in Market Simulations*. arXiv: 2504.10789. URL: <https://arxiv.org/abs/2504.10789>. Pre-published.
- Lopez-Lira, Alejandro and Yuehua Tang (Apr. 6, 2023). “Can ChatGPT Forecast Stock Price Movements? Return Predictability and Large Language Models”. In: *SSRN Electronic Journal*. ISSN: 1556-5068. DOI: [10.2139/SSRN.4412788](https://doi.org/10.2139/SSRN.4412788). URL: <https://papers.ssrn.com/abstract=4412788> (visited on 04/22/2023).
- Ludwig, Jens and Sendhil Mullainathan (May 1, 2024). “Machine Learning as a Tool for Hypothesis Generation”. In: *The Quarterly Journal of Economics* 139.2, pp. 751–827. ISSN: 0033-5533. DOI: [10.1093/qje/qjad055](https://doi.org/10.1093/qje/qjad055). URL: <https://doi.org/10.1093/qje/qjad055> (visited on 03/13/2026).
- Ludwig, Jens, Sendhil Mullainathan, and Ashesh Rambachan (Jan. 2025). *Large Language Models: An Applied Econometric Framework*. DOI: [10.3386/w33344](https://www.nber.org/papers/w33344). National Bureau of Economic Research: 33344. URL: <https://www.nber.org/papers/w33344> (visited on 04/07/2025). Pre-published.
- Luo, Xiaoliang, Akilles Rechardt, Guangzhi Sun, Kevin K. Nejad, Felipe Yáñez, Bati Yilmaz, Kangjoo Lee, Alexandra O. Cohen, Valentina Borghesani, Anton Pashkov, Daniele Marinazzo, Jonathan Nicholas, Alessandro Salatiello, Ilia Sucholutsky, Pasquale Minervini, Sepehr Razavi, Roberta Rocca, Elkhan Yusifov, Tereza Okalova, Nianlong Gu, Martin Ferianc, Mikail Khona, Kaustubh R. Patil, Pui-Shee Lee, Rui Mata, Nicholas E. Myers, Jennifer K. Bizley, Sebastian Musslick, Isil Poyraz Bilgin, Guiomar Niso, Justin M. Ales, Michael Gaebler, N. Apurva Ratan Murty, Leyla Loued-Khenissi, Anna Behler, Chloe M. Hall, Jessica Dafflon, Sherry Dongqi Bao, and Bradley C. Love (Feb. 2025). “Large Language Models Surpass Human Experts in Predicting Neuroscience

- Results". In: *Nature Human Behaviour* 9.2, pp. 305–315. ISSN: 2397-3374. DOI: [10.1038/s41562-024-02046-9](https://doi.org/10.1038/s41562-024-02046-9). URL: <https://www.nature.com/articles/s41562-024-02046-9> (visited on 03/13/2026).
- Nerkar, Atul (Feb. 2003). "Old Is Gold? The Value of Temporal Exploration in the Creation of New Knowledge". In: *Management Science* 49.2, pp. 211–229. ISSN: 0025-1909. DOI: [10.1287/mnsc.49.2.211.12747](https://doi.org/10.1287/mnsc.49.2.211.12747). URL: <https://pubsonline.informs.org/doi/10.1287/mnsc.49.2.211.12747> (visited on 03/13/2026).
- Novikov, Alexander, Ngàn Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog (June 16, 2025). *AlphaE-volve: A Coding Agent for Scientific and Algorithmic Discovery*. DOI: [10.48550/arXiv.2506.13131](https://doi.org/10.48550/arXiv.2506.13131). arXiv: [2506.13131 \[cs\]](https://arxiv.org/abs/2506.13131). URL: <http://arxiv.org/abs/2506.13131> (visited on 03/13/2026). Pre-published.
- Novy-Marx, Robert and Mihail Velikov (Mar. 2026). "Artificial Intelligence–Powered (Finance) Scholarship". In: *Journal of Economic Literature* 64.1, pp. 5–37. ISSN: 0022-0515. DOI: [10.1257/jel.20251821](https://doi.org/10.1257/jel.20251821). URL: <https://www.aeaweb.org/articles?id=10.1257/jel.20251821> (visited on 03/13/2026).
- OpenAI et al. (Aug. 8, 2025). *Gpt-Oss-120b & Gpt-Oss-20b Model Card*. DOI: [10.48550/arXiv.2508.10925](https://doi.org/10.48550/arXiv.2508.10925). arXiv: [2508.10925 \[cs\]](https://arxiv.org/abs/2508.10925). URL: <http://arxiv.org/abs/2508.10925> (visited on 03/13/2026). Pre-published.
- Ottonello, Pablo, Wenting Song, and Sebastian Sotelo (Sept. 2024). *An Anatomy of Firms' Political Speech*. DOI: [10.3386/w32923](https://doi.org/10.3386/w32923). National Bureau of Economic Research: [32923](https://www.nber.org/papers/w32923). URL: <https://www.nber.org/papers/w32923> (visited on 03/13/2026). Pre-published.
- Park, Michael, Erin Leahey, and Russell J. Funk (Jan. 2023). "Papers and Patents Are Becoming Less Disruptive over Time". In: *Nature* 613.7942, pp. 138–144. ISSN: 1476-4687.

DOI: [10.1038/s41586-022-05543-x](https://doi.org/10.1038/s41586-022-05543-x). URL: <https://www.nature.com/articles/s41586-022-05543-x> (visited on 03/13/2026).

Romera-Paredes, Bernardino, 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 (Jan. 2024). “Mathematical Discoveries from Program Search with Large Language Models”. In: *Nature* 625.7995, pp. 468–475. ISSN: 1476-4687. DOI: [10.1038/s41586-023-06924-6](https://doi.org/10.1038/s41586-023-06924-6). URL: <https://www.nature.com/articles/s41586-023-06924-6> (visited on 03/13/2026).

Runco, Mark A. and Garrett J. Jaeger (2012). “The Standard Definition of Creativity”. In: *Creativity Research Journal* 24.1, pp. 92–96. ISSN: 1532-6934. DOI: [10.1080/10400419.2012.650092](https://doi.org/10.1080/10400419.2012.650092).

Simonton, Dean Keith (1997). “Creative Productivity: A Predictive and Explanatory Model of Career Trajectories and Landmarks”. In: *Psychological Review* 104.1, pp. 66–89. ISSN: 1939-1471. DOI: [10.1037/0033-295X.104.1.66](https://doi.org/10.1037/0033-295X.104.1.66).

Teplitskiy, Misha, Hao Peng, Andrea Blasco, and Karim R. Lakhani (Nov. 22, 2022). “Is Novel Research Worth Doing? Evidence from Peer Review at 49 Journals”. In: *Proceedings of the National Academy of Sciences* 119.47, e2118046119. DOI: [10.1073/pnas.2118046119](https://doi.org/10.1073/pnas.2118046119). URL: <https://www.pnas.org/doi/10.1073/pnas.2118046119> (visited on 03/13/2026).

Tranchero, Matteo, Cecil-Francis Brenninkmeijer, Arul Murugan, and Abhishek Nagaraj (Oct. 2024). *Theorizing with Large Language Models*. DOI: [10.3386/w33033](https://doi.org/10.3386/w33033). National Bureau of Economic Research: 33033. URL: <https://www.nber.org/papers/w33033> (visited on 12/26/2024). Pre-published.

Trinh, Trieu H., Yuhuai Wu, Quoc V. Le, He He, and Thang Luong (Jan. 2024). “Solving Olympiad Geometry without Human Demonstrations”. In: *Nature* 625.7995, pp. 476–

482. ISSN: 1476-4687. DOI: [10.1038/s41586-023-06747-5](https://doi.org/10.1038/s41586-023-06747-5). URL: <https://www.nature.com/articles/s41586-023-06747-5> (visited on 03/13/2026).
- Uzzi, Brian, Satyam Mukherjee, Michael Stringer, and Ben Jones (Oct. 25, 2013). “Atypical Combinations and Scientific Impact”. In: *Science* 342.6157, pp. 468–472. DOI: [10.1126/science.1240474](https://doi.org/10.1126/science.1240474). URL: <https://www.science.org/doi/10.1126/science.1240474> (visited on 03/13/2026).
- Van Binsbergen, Jules, Michael Brandt, and Ralph Koijen (June 2012). “On the Timing and Pricing of Dividends”. In: *American Economic Review* 102.4, pp. 1596–1618. ISSN: 0002-8282. DOI: [10.1257/aer.102.4.1596](https://doi.org/10.1257/aer.102.4.1596). URL: <https://www.aeaweb.org/articles?id=10.1257/aer.102.4.1596> (visited on 03/13/2026).
- Wang, Jian, Reinhilde Veugelers, and Paula Stephan (Oct. 1, 2017). “Bias against Novelty in Science: A Cautionary Tale for Users of Bibliometric Indicators”. In: *Research Policy* 46.8, pp. 1416–1436. ISSN: 0048-7333. DOI: [10.1016/j.respol.2017.06.006](https://doi.org/10.1016/j.respol.2017.06.006). URL: <https://www.sciencedirect.com/science/article/pii/S0048733317301038> (visited on 03/13/2026).
- Wu, Jing Cynthia, Jin Xi, and Shihan Shihan. Xie@gmail.com (Oct. 1, 2025). *LLM Survey Framework: Coverage, Reasoning, Dynamics, Identification*. Social Science Research Network: 5665312. URL: <https://papers.ssrn.com/abstract=5665312> (visited on 03/13/2026). Pre-published.
- Yu, Yulin and Daniel M. Romero (Oct. 8, 2024). “Does the Use of Unusual Combinations of Datasets Contribute to Greater Scientific Impact?” In: *Proceedings of the National Academy of Sciences* 121.41, e2402802121. DOI: [10.1073/pnas.2402802121](https://doi.org/10.1073/pnas.2402802121). URL: <https://www.pnas.org/doi/10.1073/pnas.2402802121> (visited on 03/13/2026).

# Appendices

## A Prompts Used in the System

The system uses the following prompts for theory generation.

### A.1 Problem analyzer prompts

#### System prompt

- You are an expert theoretical economist analyzing empirical puzzles. Your task is to deeply understand why standard models fail to explain  
→ observed data, and identify promising directions for new  
→ theoretical explanations.  
  
THINK REALLY HARD. This is a genuine research problem that has stumped  
→ economists.  
  
Take your time to reason through the mechanics carefully.  
Think like a researcher preparing to write a theory paper:
  - What are the key facts that need explaining?
  - Why EXACTLY do existing models fail? Be mechanistic, not hand-wavy.
  - What assumptions might be wrong? Which ones matter most?
  - What novel mechanisms could explain the data? Be specific about  
→ functional forms.Be specific and analytical. Avoid vague suggestions like "add  
→ frictions" - instead say  
  
exactly what kind of friction and how it would affect the equilibrium.

- **Analysis prompt template:** The template below is filled with puzzle-specific content; placeholders {puzzle\_description}, {targets\_description}, {baseline\_description}, and {additional\_context} are substituted at runtime.

## Analysis prompt template

### # Task

Think really hard about this. Deeply analyze the following empirical  
→ puzzle.

Your analysis will guide the development of new theoretical models to  
→ explain the data.

This is a real research problem - your insights matter. Don't rush.

# The Puzzle {puzzle\_description}

# Empirical Targets {targets\_description}

# Baseline Model {baseline\_description}

# Additional Context {additional\_context}

# Your Analysis

Provide a thorough analysis addressing:

1. **\*\*Empirical Facts\*\***: What are the 2-4 key facts that need  
→ explaining?
2. **\*\*Baseline Failure\*\***: Why exactly does the baseline model fail? Be  
→ mechanistic.
3. **\*\*Violated Assumptions\*\***: Which assumptions of the baseline are  
→ likely wrong?
4. **\*\*Promising Directions\*\***: What are 3-5 theoretical directions worth  
→ exploring?
  - Be specific (not just "add frictions" but "transaction costs that  
→ scale with...")
  - Consider: behavioral biases, market frictions, heterogeneous  
→ agents,  
information asymmetry, institutional constraints, etc.
5. **\*\*Dead Ends\*\***: What approaches are unlikely to work? Why?

6. **\*\*Key Constraints\*\***: What must any successful theory satisfy?

- Mathematical constraints
- Economic plausibility requirements
- Parsimony considerations

7. **\*\*Unconventional Idea\*\***: Suggest one surprising or unconventional  
 ↳ angle that  
     might work but isn't obvious.

Respond in the structured JSON format.

## A.2 Theory evolution prompts

### System prompt

- You are an expert theoretical economist specializing in asset pricing. You propose novel theoretical models to explain empirical puzzles. You think creatively but rigorously, and your models are mathematically  
 ↳ precise.

CRITICAL: Parsimony is paramount. You are fitting  $N$  target moments.

- ↳ Models with more than  $N$  free parameters are overparameterized and
- ↳ will be heavily penalized. A simple model that approximately hits
- ↳ targets is far better than a complex model that fits exactly.

CRITICAL: Before elaborating your model, verify that your proposed

- ↳ mechanism can mathematically achieve the target moments with
- ↳ reasonable parameter values. A model that cannot hit the targets
- ↳ regardless of calibration will fail.

- **Random strategy**: Propose a completely new theory. Placeholders {puzzle\_context}

and {previous\_attempts} are filled at runtime.

### Random strategy

```
# Task

Propose a completely novel theoretical model to explain the empirical
→ puzzle.

---

{puzzle_context}

---

# Previous Attempts

{previous_attempts}

---

# Your Task

Propose a NEW theoretical model that matches the empirical targets and
→ is DIFFERENT from previous attempts.

Provide your response as JSON with these fields:

- model_name: Short descriptive name (e.g., "Risk-Information Hybrid
→ Model")

- key_insight: 1-2 sentence core insight

- formulas: Dictionary mapping each output name to its mathematical
→ formula

- parameters: List of parameters with economic interpretation (fewer is
→ better)

- mechanism_explanation: Why this mechanism produces target values

- full_proposal: Complete detailed proposal with all derivations
```

- **Mutate strategy:** Modify a successful parent model. Placeholders include {parent\_name}, {parent\_insight}, {parent\_formulas}, {parent\_params}, {parent\_score}, {parent\_results}, and referee feedback fields.

## Mutate strategy

```
# Task

Take this successful model and propose a VARIATION that might work even
→ better.

---

# Parent Model: {parent_name}

## Key Insight

{parent_insight}

## Formulas

{parent_formulas}

## Parameters ({n_params} free parameters)

{parent_params}

## Results

- Score: {parent_score}/100

{parent_results}

## Referee Feedback

**Strengths:** {referee_strengths}

**Weaknesses:** {referee_weaknesses}

**Suggestions:** {referee_suggestions}

---

# Mutation Instructions

Propose a modification to this model. Options (in order of preference):

1. **Simplify** - Remove a parameter or combine terms (STRONGLY
→ PREFERRED - improves parsimony)

2. **Modify mechanism** - Change how a parameter enters the model

3. **Add interaction** - Combine existing parameters in a new way

4. **Add new effect** - Introduce a new economic friction (DISCOURAGED
→ unless essential)
```

The referee noted weaknesses above. Your mutation should address at  
↪ least one.

The mutation should:

- Keep the core insight but modify the mechanism
- Be economically plausible
- Address referee concerns
- Improve parsimony if possible

---

# Your Task

Provide your response as JSON with these fields:

- model\_name: Short descriptive name for the mutated model
- key\_insight: 1-2 sentence core insight (what changed and why)
- formulas: Dictionary with new mathematical formulas for each output
- parameters: List of parameters (note which are new/removed)
- mechanism\_explanation: Why this mutation improves the model
- full\_proposal: Complete detailed proposal with derivations

- **Crossover strategy:** Combine two models into a hybrid. Placeholders {model\_a\_name}, {model\_a\_insight}, {model\_a\_formulas}, {model\_a\_score}, and similarly for model B, are substituted at runtime.

### Crossover strategy

# Task

Combine ideas from these two models to create a hybrid.

---

# Model A: {model\_a\_name}

## Key Insight

{model\_a\_insight}

```

## Formulas
{model_a_formulas}

## Score: {model_a_score}/100

---

# Model B: {model_b_name}

## Key Insight
{model_b_insight}

## Formulas
{model_b_formulas}

## Score: {model_b_score}/100

---

# Your Task

Create a hybrid model combining the best elements of both. AIM FOR
→ FEWER total parameters than the sum - find overlap!

Provide your response as JSON with these fields:

- model_name: Short descriptive name for the hybrid model
- key_insight: 1-2 sentence core insight (what elements are combined)
- formulas: Dictionary with new mathematical formulas for each output
- parameters: Combined parameter list (aim for fewer than sum of both)
- mechanism_explanation: Why this combination works better than either
→ alone
- full_proposal: Complete detailed proposal with derivations

```

The *mutate* prompt includes the parent model’s full referee report (strengths, weaknesses, suggestions), so mutation incorporates referee-driven refinement directly. There is no separate “refine” strategy; referee feedback is integrated into the mutate operator.

## A.3 Coding agent prompts

### System prompt

- You are an expert Python programmer specializing in mathematical  
→ modeling.  
Your task is to translate theoretical economic models into executable  
→ Python code.

IMPORTANT RULES:

1. Implement the formulas EXACTLY as written in the proposal
2. Do NOT use any specific numerical values claimed by the proposer
3. All parameters should be inputs to the function, not hardcoded
4. Include clear comments mapping code to the mathematical formulas
5. Handle edge cases (division by zero, negative values, etc.)
6. Do NOT use "from \_\_future\_\_ import annotations" - use standard type  
→ hints

You write clean, well-documented code with type hints.

- **User prompt template:** The placeholder {model\_proposal} is substituted with the theoretical proposal at runtime.

### User prompt template

```
# Task  
  
Translate the following theoretical model into executable Python code.  
  
---  
  
# Model Proposal  
  
{model_proposal}  
  
---
```

```

# Required Interface

You MUST implement this exact interface:

```python
from dataclasses import dataclass
from typing import Dict, Any

@dataclass
class ModelParams:
    """Parameters for the model."""

    # Standard calibration parameters (from the paper)
    N: int = 2278          # total number of stocks
    S: int = 9             # number of style portfolios
    N_s: int = 253         # stocks per style (N/S)
    sigma_style: float = 0.104 # style shock volatility (annual)
    sigma_idio: float = 0.435  # idiosyncratic volatility (annual)
    sigma_market: float = 0.16 # market volatility (annual)

    # ADD ANY MODEL-SPECIFIC PARAMETERS HERE with sensible defaults

def compute_multipliers(params: ModelParams) -> Dict[str, Any]:
    """
    Compute price multipliers M_style and M_idio for this model.
    Returns:
        Dictionary with keys: M_style, M_idio, ratio, intermediates
    """

    # YOUR IMPLEMENTATION HERE

    pass
```

```

```

# Instructions

1. Add any new parameters from the model proposal to ModelParams
2. Implement compute_multipliers() based on the mathematical formulas
3. Include comments showing which formula each line implements
4. Return intermediate values for debugging
5. Do NOT hardcode any numbers - use params.*

---

# Output Format

Return ONLY valid Python code. No explanation before or after.

The code must be directly executable.

```python
# Your complete implementation here
```

```

## A.4 Calibration agent prompts

### System prompt

- You are an expert econometrician who calibrates theoretical models.

Given a theoretical model proposal and its Python implementation, you

→ determine:

1. Sensible default values for each parameter
2. Valid bounds for optimization (min, max)
3. Test values for robustness checking
4. Parameter types (continuous, count, fraction)

You understand economic constraints:

- Risk aversion ( $\gamma$ ) is typically 1-10

- Counts (K, n\_stocks) must be  $\geq 1$  and often  $\leq N$  (total stocks)
- Fractions (delta, phi, eta) must be in  $[0, 1]$
- Costs are typically small positive numbers

Be conservative with bounds - too wide is better than too narrow.

- **Calibration prompt template:** Placeholders {proposal} and {code} are substituted with the model proposal and generated Python code at runtime.

#### Calibration prompt template

```
# Task
Analyze this model and specify calibration for each parameter.
---
# Model Proposal
{proposal}
---
# Generated Code
```python
{code}
```
---
# Fixed Calibration Parameters (from the literature)
These are FIXED and should NOT be included in your output:
- N = 2278 (total stocks)
- S = 9 (style portfolios)
- N_s = 253 (stocks per style)
- sigma_style = 0.104 (style volatility)
- sigma_idio = 0.435 (idiosyncratic volatility)
```

```

- sigma_market = 0.16 (market volatility)

---

# Your Task

For each MODEL-SPECIFIC parameter (NOT the fixed ones above), provide:

1. name: exact name from the code
2. default: sensible starting value
3. min_bound, max_bound: valid range for optimization
4. test_values: 3-5 values spanning the range (for robustness testing)
5. param_type: "continuous", "count", or "fraction"
6. description: economic interpretation

IMPORTANT:

- Only include parameters that appear in ModelParams (not the fixed
  ↳ calibration ones)
- Bounds should be economically sensible (e.g.,  $K \leq N$ , fractions in
  ↳  $[0,1]$ )
- Test values should include edge cases where appropriate
- If a parameter represents a count, min_bound should be  $\geq 1$ 

```

## A.5 Mathematical auditor prompts

The mathematical auditor verifies derivation correctness before code translation.

### Audit system prompt

- You are a mathematical economist who audits derivations for
  - ↳ correctness.
 Your job is to check every step of a theoretical proposal. Be rigorous
  - ↳ and precise.

You must verify:

1. Each equation follows from the previous one (no skipped steps)
2. Matrix algebra is correct (dimensions, inverses, transposes)
3. First-order conditions are correctly derived from the objective
4. Market clearing conditions are properly imposed
5. Final pricing formulas actually follow from the derivation

Do NOT check calibrated numerical values - those are irrelevant.

Focus ONLY on whether the mathematical logic is sound.

- **Audit prompt:** The placeholder {proposal} is substituted with the model proposal. The auditor must end its response with either OVERALL: PASS or OVERALL: FAIL.

#### Fix system prompt

- You are a theoretical economist who fixes mathematical errors in model  
→ derivations.  
You will receive a proposal and a mathematical audit identifying  
→ errors.  
Fix ONLY the errors while preserving the economic mechanism and model  
→ structure.  
Output the complete corrected proposal in the same format as the  
→ original.  
Do NOT change the economic mechanism - only fix the math.

- **Fix prompt:** Receives the original proposal and the audit feedback. The fix-and-reaudit cycle repeats up to nine times (Appendix C.3).

## A.6 Pseudocode agent prompts

The pseudocode agent translates audited proposals into structured pseudocode.

### System prompt

- You are a theoretical economist writing structured pseudocode for asset  
→ pricing models.

Your job is to translate a model proposal into STRUCTURED PSEUDOCODE

→ that a programmer

can implement exactly. The pseudocode must capture the complete

→ derivation chain from

economic primitives to numerical outputs.

CRITICAL RULES:

1. Every output must be DERIVED from the model's economic equations
2. No fudge factors, scaling parameters, or tuning knobs that directly  
→ encode target values
3. All returns, risk premia, Sharpe ratios, alphas, and betas must  
→ emerge from the  
model's pricing kernel / stochastic discount factor (SDF)
4. Conditional moments must arise from the model's state variable  
→ dynamics
5. Be FAITHFUL to the proposal - include every equation and mechanism  
→ mentioned
6. Be COMPLETE - every target output must have a clear derivation path

The pseudocode must make the full chain visible:

Economic primitives -> SDF/pricing kernel -> Asset prices -> Returns

→ -> Target outputs

- **User prompt:** Receives the proposal and a list of target outputs. The pseudocode must be organized into four sections: PARAMETERS, DERIVATION, COMPUTE OUTPUTS, and CONDITIONAL MOMENTS.

## A.7 Pseudocode reviewer prompts

The pseudocode reviewer verifies that pseudocode faithfully captures the proposal.

### System prompt

- You are reviewing pseudocode against a theoretical model proposal. Your job is to verify that the pseudocode faithfully captures the
  - model's economic mechanism and derivation chain.

PASS the pseudocode if: the core economic mechanism is preserved, all
 
  - key equations appear in the derivation, all target outputs are derived from model
    - equations.

FAIL the pseudocode if: fudge parameters scale outputs toward targets;
 
  - outputs are stated but not derived from the SDF; the pricing kernel does not match
    - the proposal;
  - the core economic channel is absent; conditional moments are assumed
    - rather than derived.

Key principle: Every numerical output must be traceable back to the SDF
 
  - and model parameters through a chain of economic equations.

- **Review prompt:** Receives the original proposal and the generated pseudocode. Returns a PASS/FAIL verdict with a list of issues and a one-sentence summary. If the review fails, the pseudocode agent regenerates with the reviewer’s feedback (up to two retries).

## A.8 Code-vs-pseudocode reviewer prompts

The code-vs-pseudocode reviewer verifies that generated Python code implements the pseudocode specification.

### System prompt

- You are reviewing Python code against its pseudocode specification for a theoretical asset pricing model.

Check if the code faithfully implements the pseudocode:

1. PARAMETERS: Are all pseudocode parameters present as function  
→ arguments?
2. DERIVATION: Does the code implement the SDF / pricing kernel  
→ equations?
3. OUTPUTS: Does the code compute each target output using the derived  
→ formulas?
4. CONDITIONAL MOMENTS: If the pseudocode specifies state-dependent  
→ computation,  
does the code implement it?

PASS the code if the core formulas match the pseudocode.

FAIL the code only for CRITICAL mismatches: extra fudge parameters,  
→ wrong formulas,  
outputs hardcoded rather than derived, or missing key derivation steps.

Key principle: The code must compute outputs through the same  
 $\hookrightarrow$  derivation chain as  
the pseudocode. If an output bypasses the SDF, FAIL.

- **Review prompt:** Receives the pseudocode specification and the generated Python code. Returns a PASS/FAIL verdict with issues and summary. If the review fails, the coding agent regenerates with feedback; a review that still fails after the retry *rejects* the run (it is recorded with `stage_failed=code_review` and never scored).

## A.9 Evaluator: robustness check and optimization

The evaluator runs in Python (no LLM) and performs two steps.

**Robustness check.** The calibration agent supplies test values for each model-specific parameter. The evaluator builds a list of parameter combinations (e.g., default, plus each test value varied in turn) and runs `compute_multipliers(params)` for each. The check passes only if every run completes without exception and all outputs are finite. This check catches division-by-zero, invalid domains, and NaN/Inf.

**Parameter optimization.** The objective is to minimize the sum of relative squared errors to the empirical targets. Each puzzle defines its own outputs and targets. For the price-multiplier puzzle, the fitted outputs are the three multiplier levels  $M_{\text{style}}$ ,  $M_{6 \times 6}$ , and  $M_{\text{idio}}$  (the ratio is a derived diagnostic). For the dividend strip puzzle, the eight independent outputs are fitted; for the currency puzzle, four. In general, for outputs  $y_1, \dots, y_k$  with targets  $t_1, \dots, t_k$ , the objective is

$$L(x) = \sum_{i:t_i \neq 0} \left( \frac{y_i(x) - t_i}{t_i} \right)^2, \quad (16)$$

where  $x$  is the vector of model-specific parameters. All outputs receive equal weight. The

scorer uses the same (equal-weighted) average percentage error when computing the fit subscore, so the objective and fit score are consistent. The evaluator uses `scipy.optimize.differential` with bounds from the calibration agent (or defaults such as  $\gamma \in [0.1, 10]$ , fractions in  $[0, 1]$ ), with `maxiter=100`, `tol=1e-6`, `polish=True`, and `seed=42`. The optimizer stage is intentionally lenient: a run is discarded only if the solver fails to converge *and* the final loss exceeds  $L > 5.0$ , which corresponds to very large errors. All other models, including those with moderate fit, proceed to the scorer and referee, which judge quality on a continuous scale. The optimal parameters and resulting outputs (e.g.,  $M_{\text{style}}$ ,  $M_{\text{idio}}$ , ratio) are returned for the scorer and referee.

## A.10 Scorer: formulas

The scorer combines three subscores with configurable weights. In the reported experiment we use  $w_{\text{fit}} = 0.50$ ,  $w_{\text{plaus}} = 0.25$ ,  $w_{\text{pars}} = 0.25$ . Denote the weight for dimension  $d$  by  $w_d$  with  $\sum_d w_d = 1$ .

**Fit.** For each output  $i$  with value  $v_i$  and target  $t_i \neq 0$ , the percentage error is  $e_i = 100 \cdot |v_i - t_i| / |t_i|$ . The average error is the simple (equal-weighted) mean:  $\bar{e} = (\sum_i e_i) / k$ , where  $k$  is the number of outputs with non-zero targets. The fit score is

$$\text{Fit} = \min(100, \max(0, 100 e^{-\bar{e}/20})). \quad (17)$$

So perfect fit gives 100; score decays exponentially with average percentage error (e.g.,  $\bar{e} \approx 20\%$  gives about 37).

**Plausibility.** Each calibrated parameter is checked against predefined bounds (e.g., risk aversion  $\gamma \in [0.5, 5]$ , fractions in  $[0, 1]$ ). Let  $n$  be the number of parameters checked and  $m$  the number within bounds. The base score is  $(m/n) \times 100$ . If any parameter is out of bounds, a penalty of up to 30 points is applied based on the maximum deviation; the final plausibility score is  $\max(0, \text{base} - \text{penalty})$ .

**Parsimony.** The parsimony score is identification-aware. Let  $n$  be the number of free (model-specific) parameters and  $M$  the number of independent moments the puzzle declares for fitting. Then

$$\text{Parsimony} = \begin{cases} \max(56, 100 - 8n) & n < M \quad (\text{over-identified: the genuine result}), \\ 55 & n = M \quad (\text{just-identified: fit is automatic}), \\ \max(0, 55 - 25(n - M)) & n > M \quad (\text{under-identified: fit is contentless}). \end{cases} \quad (18)$$

An over-identified model—fewer parameters than fitted moments—is the prize, because hitting every moment is then a genuine test; a just-identified fit is guaranteed by construction and docked; an under-identified model gives the optimizer slack to launder a fit and is penalized steeply. Every puzzle in this paper declares its independent fit-moment count explicitly.

**Total.**  $\text{Total} = w_{\text{fit}} \cdot \text{Fit} + w_{\text{plaus}} \cdot \text{Plausibility} + w_{\text{pars}} \cdot \text{Parsimony}$ , rounded to one decimal.

## A.11 Referee prompts

### System prompt

- You are a senior referee for a top economics journal.

You evaluate theoretical model proposals that attempt to explain  
 $\rightarrow$  empirical puzzles.

You judge based on:

1. Empirical fit - does it match the data?
2. Economic plausibility - does the mechanism make sense?
3. Novelty - is this a new contribution?
4. Parsimony - is it elegantly simple?

5. Implementation quality - is the math correctly implemented?

You provide constructive, specific feedback that would help improve the  
→ paper.

You are fair but rigorous.

- **Referee prompt template:** Placeholders {proposal}, {scoring\_section}, {quant\_total}, {consistent}, {consistency\_verdict}, {discrepancies}, and {optimization\_results} are filled with the model proposal, quantitative scores breakdown, weighted total, consistency result, and optimization summary. The template also contains a *Mechanism Audit* block, executed before scoring: for each factor, shock, or parameter the proposal's narrative claims drives a target moment, the referee checks whether that symbol appears in the derived formula for that moment, flagging decorative ingredients and state-dependence claims whose formulas are algebraically identical across states; and a *Deterministic Code Scan* section that surfaces any warnings raised by the pipeline's static code checks (e.g., a literal suspiciously close to a target). The referee must score the artifact *as submitted*: it may suggest repairs in prose but may not score a hand-repaired counterfactual. The referee returns structured JSON with: model\_name, score (0–100), recommendation (reject / major\_revision / minor\_revision / accept), summary, strengths, weaknesses, suggestions, novelty\_assessment, economic\_plausibility, reasoning. Scoring guidelines: 90–100 excellent; 75–89 good, minor revisions; 60–74 promising, major revisions; 40–59 significant issues; 0–39 reject. The recorded referee score is the *median* of three independent referee calls; all three reports are stored and fed back to the evolver.

### Referee prompt template

(Full template omitted; placeholders and required JSON schema as described in the preceding paragraph.)

## B Derivation of the Benchmark Price Multiplier Ratio

This appendix derives from first principles the benchmark ratio  $M_{\text{style}}/M_{\text{idio}} = N_s \sigma_{\text{style}}^2 / \sigma_{\text{idio}}^2$ . Under the calibrated inputs used in the main text, it equals 14.5.

### B.1 Setup: CARA-Normal liquidity provider and the multiplier

Let  $r \in \mathbb{R}^{N_s}$  denote the vector of (excess) returns of the  $N_s$  stocks in one style bucket, and assume  $r \sim \mathcal{N}(\mu, \Sigma)$ . A representative (marginal) risk-averse investor with CARA-Normal mean–variance preferences chooses a holdings vector  $x \in \mathbb{R}^{N_s}$  to maximize

$$\mu^\top x - \frac{\gamma}{2} x^\top \Sigma x, \quad (19)$$

where  $\gamma > 0$  is the coefficient of absolute risk aversion. The first-order condition is  $\mu = \gamma \Sigma x$ . Suppose an order-flow shock forces the market to absorb an exogenous net supply  $q$  (e.g., flow-induced inventory). Market clearing for the marginal absorber implies  $x = q$ , so

$$\mu = \gamma \Sigma q. \quad (20)$$

Thus the expected return (risk premium) required for the marginal investor to hold  $q$  is proportional to the risk of that position. A “price multiplier” is the amount of expected return or discount required per unit of flow shock. In this one-period benchmark, the mapping from  $q$  to  $\mu$  is linear with coefficient proportional to  $\gamma \Sigma$ ; up to a common duration or price-elasticity constant that cancels in ratios, the multiplier is proportional to  $\gamma \Sigma$ .

## B.2 Covariance structure: style and idiosyncratic components

Each stock's return is written as  $r_i = s + \varepsilon_i$ , where  $s$  is the style common shock with  $\text{Var}(s) = \sigma_{\text{style}}^2$ , and  $\varepsilon_i$  is the idiosyncratic component with  $\text{Var}(\varepsilon_i) = \sigma_{\text{idio}}^2$ . The  $\varepsilon_i$  are mutually independent and independent of  $s$ . The covariance matrix is therefore

$$\Sigma = \sigma_{\text{style}}^2 \mathbf{1}\mathbf{1}^\top + \sigma_{\text{idio}}^2 I, \quad (21)$$

where  $\mathbf{1}$  is the  $N_s \times 1$  vector of ones. Hence  $\text{Cov}(r_i, r_j) = \sigma_{\text{style}}^2$  for  $i \neq j$ , and  $\text{Var}(r_i) = \sigma_{\text{style}}^2 + \sigma_{\text{idio}}^2$  for each  $i$ .

## B.3 Idiosyncratic-shock multiplier $M_{\text{idio}}$

An idiosyncratic flow shock is a supply shock in a single stock:  $q^{\text{idio}} = e_k$ , where  $e_k$  is the unit vector with 1 in position  $k$  and 0 elsewhere. Substituting into (20) gives  $\mu = \gamma \Sigma e_k$ . The  $k$ -th component is  $\mu_k = \gamma \Sigma_{kk} = \gamma(\sigma_{\text{style}}^2 + \sigma_{\text{idio}}^2)$ . Following Li and Lin (2026), the idiosyncratic multiplier is defined to isolate the idiosyncratic variance contribution (the part due to  $\varepsilon_k$ ), i.e.,

$$M_{\text{idio}} \equiv \gamma \sigma_{\text{idio}}^2. \quad (22)$$

## B.4 Style-level shock multiplier $M_{\text{style}}$

A style-level flow shock is a supply shock that hits all  $N_s$  stocks in the same direction, e.g., one unit in each:  $q^{\text{style}} = \mathbf{1}$ . Substituting into (20) yields

$$\mu = \gamma \Sigma \mathbf{1} = \gamma \left( \sigma_{\text{style}}^2 \mathbf{1}\mathbf{1}^\top \mathbf{1} + \sigma_{\text{idio}}^2 I \mathbf{1} \right). \quad (23)$$

Since  $\mathbf{1}^\top \mathbf{1} = N_s$ , one has  $\mathbf{1}\mathbf{1}^\top \mathbf{1} = N_s \mathbf{1}$ , and  $I \mathbf{1} = \mathbf{1}$ . Thus

$$\mu = \gamma \left( N_s \sigma_{\text{style}}^2 \mathbf{1} + \sigma_{\text{idio}}^2 \mathbf{1} \right) = \gamma \left( N_s \sigma_{\text{style}}^2 + \sigma_{\text{idio}}^2 \right) \mathbf{1}, \quad (24)$$

so each stock's required premium is  $\mu_i = \gamma(N_s\sigma_{\text{style}}^2 + \sigma_{\text{idio}}^2)$ . Similarly, the style multiplier isolates the systematic scaling term:

$$M_{\text{style}} \equiv \gamma N_s \sigma_{\text{style}}^2. \quad (25)$$

## B.5 The ratio

With these definitions,

$$\frac{M_{\text{style}}}{M_{\text{idio}}} = \frac{\gamma N_s \sigma_{\text{style}}^2}{\gamma \sigma_{\text{idio}}^2} = \frac{N_s \sigma_{\text{style}}^2}{\sigma_{\text{idio}}^2}. \quad (26)$$

## B.6 Numerical value 14.5

Using the calibrated inputs  $N_s = 253$ ,  $\sigma_{\text{style}} = 0.104$ , and  $\sigma_{\text{idio}} = 0.435$ , one obtains  $\sigma_{\text{style}}^2 = 0.010816$  and  $\sigma_{\text{idio}}^2 = 0.189225$ , so

$$\frac{M_{\text{style}}}{M_{\text{idio}}} = \frac{253 \times 0.010816}{0.189225} = \frac{2.736448}{0.189225} \approx 14.5. \quad (27)$$

The factor  $N_s$  in the numerator arises from  $\mathbf{1}\mathbf{1}^\top \mathbf{1} = N_s \mathbf{1}$ : a correlated (style) position loads on the common shock in every stock, so the marginal investor's required premium per stock scales with the breadth of the correlated inventory they must absorb.

# C System Implementation Details

The system runs on Python 3.12. The `llm_service.py` module provides an LLM client wrapper with retry, exponential backoff, and an outage-tolerant schedule for transient provider errors. The `evolver.py` module implements evolution strategies (random, mutate, crossover), parent selection, and the 20-persona system. The `proposal_gate.py` module implements the proposal-level cheating gate (five-vote judge panel). The `sympy_agent.py` module executes the symbolic derivation stage, including the harness-owned operators

(strip-return moments, pricing-kernel moments, the equilibrium-response comparative static) and the owned-output equality checks; `sympy_codegen.py` compiles verified formulas into Python helpers for the coding agent; `transcription_gate.py` verifies the derivation transcribed the proposal faithfully. The `math_auditor.py` module performs independent mathematical auditing with fix-and-reaudit. The `pseudocode_agent.py` and `pseudocode_reviewer.py` modules translate proposals to structured pseudocode and verify fidelity. The `coding_agent.py` module handles code translation with nested retry and the deterministic validation checks (state-realization gate, hardcoded-target and unit-conversion checks). The `code_vs_pseudocode_reviewer.py` module verifies code fidelity to the pseudocode, and `constraint_agent.py` enforces puzzle-declared solution constraints. The `calibration_agent.py` module extracts parameter bounds and defaults. The evaluator and scorer modules perform parameter optimization and quantitative scoring; `code_sympy_reconciler.py` numerically reconciles the scored code against the verified derivation at the optimum. The `referee.py` module provides panel-based simulated peer review. The `model_registry.py` module handles persistent storage of models, keyed by run seed, with lineage links and output fingerprints. The `exec_guard.py` module hard-timeouts every execution of generated code. The `orchestrator.py` module coordinates the pipeline. The `run_parallel.py` module runs batch experiments with seed tracking. The `puzzles/` package provides a modular puzzle abstraction, with each puzzle defining its own targets, calibration inputs, prompts, and harness-owned derivation requirements.

## C.1 Prompting Strategy

For theory generation, we provide the LLM with the empirical puzzle (target values, baseline model prediction), constraints (must be different from existing approaches), output format (key insight, mathematical formulas, parameters, intuition), and prior model examples and referee feedback for mutation strategies.

For code generation, we use a nested retry loop: 2 fresh starts, each with up to 3 fix

attempts. Each attempt receives feedback on syntax errors (from `ast.parse`), robustness test failures (boundary conditions), and numerical issues (NaN, Inf values). After code generation, the code-vs-pseudocode reviewer checks fidelity, with up to one retry if the review fails.

## C.2 Strategy and parent selection

Once at least one model has completed the full pipeline, strategy probabilities are: random 35%, mutate 35%, crossover 30%. Before any model completes, every run uses the random strategy. Parents for mutate and crossover are drawn from all scored models with probability proportional to the square of the referee score (no cutoff, no top- $k$  restriction); for crossover, the second parent is drawn from a different lineage than the first whenever one exists. The mutate prompt includes the parent’s full referee panel (all votes’ strengths, weaknesses, suggestions), so mutation incorporates referee feedback directly. When the proposal gate flags a proposal, the re-roll retries the same strategy and parents, so gate retries do not distort the strategy mix. The experiment uses a fixed random seed per run so that strategy and parent selection are reproducible.

## C.3 Retry and rejection policy

Table 14 summarizes the retry policy used by each component in the pipeline. A component whose retries are exhausted *rejects* the run (it is recorded with its failure stage, never scored)—rejection, not silent acceptance, is the terminal state of every gate.

Representative top-scoring models are presented in full in Section 5.2 (NFRIC and the Concave-Breadth model for price multipliers).

Table 14: Retry Configuration

| Component                                               | Type                                                   | Attempts                  |
|---------------------------------------------------------|--------------------------------------------------------|---------------------------|
| LLM API (logic errors)                                  | Exponential backoff                                    | 3 (sleeps 1s, 2s)         |
| LLM API (transient provider errors: 5xx/429/connection) | Capped backoff; SDK retries disabled                   | 8 (5s doubling, cap 180s) |
| Proposal Gate                                           | Re-roll with judges’ rationale (same strategy/parents) | 1 + 2 re-rolls            |
| Transcription Gate                                      | Re-derive on confirmed drop                            | $\leq 2$                  |
| Mathematical Auditor                                    | Audit + fix-and-reaudit                                | 1 + 9 fixes               |
| Pseudocode Agent                                        | Generate + review-and-retry                            | 1 + 2 retries             |
| Coding Agent                                            | Fresh starts $\times$ retries                          | $2 \times 3 = 6$          |
| Code-vs-Pseudocode Review                               | Review + retry, then reject                            | 1 + 1 retry               |
| Constraint Agent                                        | Regenerate with judge feedback                         | 1 + 1 retry               |
| Code-SymPy Reconciler                                   | Regenerate + re-optimize                               | 1 retry                   |
| Structured Output                                       | Simple retry                                           | 3                         |

*Notes:* Retry policies used by the production pipeline. The deep math-audit budget is a measured choice: across 7,369 audited runs, 15% of proposals pass on attempts 4–10 and score no worse than early passers, so rescuing them is cheaper than fresh runs. Every execution of LLM-generated code additionally runs under a hard (signal-based) timeout, and each run has a cooperative 30-minute deadline with a hard kill 15 minutes later, so a hung derivation or optimizer fails the run rather than the worker.

## D Scoring Methodology

The quantitative score combines three dimensions with weights 0.50 (fit), 0.25 (plausibility), 0.25 (parsimony) in the reported experiments. See Appendix A.10 for the exact formulas. Fit measures how closely the model matches empirical targets (average percentage error, exponential decay). Plausibility evaluates parameter reasonableness against predefined bounds. Parsimony is identification-aware: it rewards models with fewer free parameters than fitted moments and penalizes just- and under-identified models, whose fit is guaranteed by construction.