UNSATZ: EVALUATING SCIENTIFIC DISCOVERY WITHOUT A GIVEN ANSATZ

Anonymous authors
Paper under double-blind review

ABSTRACT

Before a scientist fits any parameter, they must decide what the model contains: which quantities exist, which are hidden, how they interact, and how they are measured. Physicists call such a trial form an *ansatz*. Benchmarks for automated equation discovery mostly supply it, by fixing the variables, a library of candidate terms, or a law to be modified. We introduce UNSATZ, a benchmark in which an LLM agent must construct the ansatz itself. Each task is a simulated system, generated from a private seed as a program in a small modeling language: ordinary, delay, or partial differential equations, discrete maps, static and causal models, or stochastic processes, possibly with hidden states, distorted sensors, and uninformative measurement channels. The agent buys experiments within a budget, writes candidate programs in the same language, obtains their coefficients from the benchmark's fitter, and is scored only on how its final program predicts experiments it never saw: conditions outside the range it could explore, interventions, and longer runs. Across 80 systems in 20 domains, nine language models solve between 21% and 60%. On 44 of these systems, classical equation-discovery methods almost never succeed, although the same data usually suffice once the true equations are given with unknown coefficients; the difficulty lies in finding the structure, not in estimating it. Because mechanisms recur across systems, we also give agents the program they discovered for a related system; it helped only when the new system shared the mechanism and the first investigation had succeeded. Stronger agents make fewer unscientific errors; beyond these, agents find which variables a system has but not how they act, and prefer revisions that lower the training error to revisions that predict better.

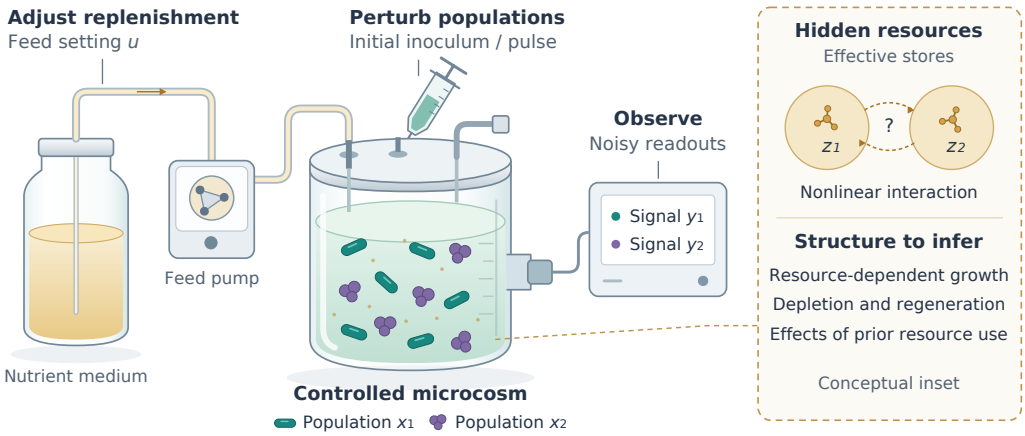
1 INTRODUCTION

Consider a scientist presented with an unfamiliar ecological microcosm (Figure 1). Two populations can be measured over time, the rate of a nutrient feed can be set, and only a limited number of runs is affordable. Before estimating a single coefficient, the scientist must decide what the model contains. Are two population variables enough, or is an unmeasured resource needed to explain why growth slows? Do the populations compete directly, or only through that resource? Does their response to the resource saturate? Experiments can answer these questions, since a pulse added to one population separates direct from resource-mediated interaction and a long run exposes slow depletion, but each experiment has to be chosen with a hypothesis in mind. Only once these decisions are made does fitting become a well-posed estimation problem. In the world of Figure 1, the answer is two hidden resources, each feeding one population and depleted by the other, so that the populations compete only through them; the agent's prompt reveals none of this.

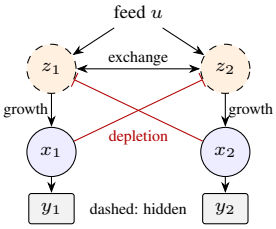
System identification calls this choice the selection of a model structure, as distinct from the estimation of its parameters (Ljung, 1999), and physicists call such a trial form an *ansatz*. We use the latter term for the structure as a whole: which variables exist, which are latent, how they interact, and how they produce the measurements. Proposing a good ansatz is central to scientific modeling, yet evaluations of automated discovery largely take it as given. Symbolic-regression benchmarks provide datasets whose variables are already chosen (Udrescu & Tegmark, 2020; La Cava et al., 2021; Shojaee et al., 2025b), and sparse identification of nonlinear dynamics (SINDy) selects equations from a fixed library of candidate functions (Brunton et al., 2016). Interactive benchmarks let the agent design experiments, but draw their targets from fixed families of probabilistic models or from modifications

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

(a) A physical interpretation



(b) Hidden structure



(c) Information available to the agent

System message (excerpt): A controlled ecological microcosm with 2 recorded population signals. You may choose the directly controllable starting populations, change the replenishment setting, apply an impulse, hold a population fixed, or purchase a wider-range run. Some recorded signals may be uninformative, and resource conditions may affect the dynamics. Budget: 243 units.

Agent → tool: `act({"action":{"type":"observe"}})`

Tool → agent:
```
{"cost": 2, "remaining": 241.0,
 "channels": ["y2", "y1"],
 "t": [0.0, 0.3333, ..., 8.0],
 "y": [[0.6738, 0.2380], ...], ...}
```

Figure 1: An ecology world. (a) A physical interpretation, for the reader. (b) The hidden structure of its true program: each population x_i grows on a store z_i that the feed u replenishes and the *other* population depletes (loss and turnover terms omitted). (c) The information available to the agent: an excerpt of its prompt and its first experiment (Appendix L).

Table 1: Benchmarks for discovering scientific models, compared with UNSATZ.

Benchmark	Data	Hidden systems	Variables	Scored on
SRBench	Fixed	Known equations, real datasets	Given	Symbolic match, test fit
LLM-SRBench	Fixed	Rewritten and synthetic equations	Given	LLM-judged match, out-of-domain error
BoxingGym	Chosen, $\leq$10	10 probabilistic models	Given	Prediction error, explanation
DiscoveryWorld	Actions, $\leq$1,000	Variants of 24 task templates	Given	Task completion, LLM-graded report
NewtonBench	Chosen, $\leq$200	Modified canonical laws	Given	LLM-judged match, data error
PhysGym	Chosen, $\leq$100	PHYBench physics problems	Given or masked	Symbolic match
Gravity-Bench	Chosen, $\leq$100	Simulated binary stars	Given	Answers within tolerance
DiscoverPhysics	Chosen, $\leq$16 rounds	Non-canonical N-body worlds	Some hidden	Held-out trajectories, LLM rubric
UNSATZ	Priced, budgeted	Generated from private seeds	Partly latent	Out-of-range, intervention, long-run tests

of canonical laws (Gandhi et al., 2025; Zheng et al., 2026), and the closest, DiscoverPhysics, asks for force laws in curated particle worlds (Wiemann et al., 2026). In each case, a person has already decided what form the answer can take, and usually which variables exist (Table 1).

The above gap matters because large language model (LLM) agents are increasingly built as research assistants that design experiments, analyze data, and propose hypotheses (Boiko et al., 2023; Lu et al., 2024; Chen et al., 2025b). Whether they can propose an ansatz when neither the variables nor the form of the answer is given is unknown, and measuring it requires a benchmark with four properties. (i) The equations that generate the system must be known to the evaluator, absent from the agent's training data, and expressible by the agent. (ii) A failure must be attributable to the agent's structure or experiments, not to the way its coefficients were estimated. (iii) A model must be judged by its predictions rather than its formula, since different formulas can describe the same system. (iv) Experiments must cost enough for their choice to matter, yet suffice for a good investigator.

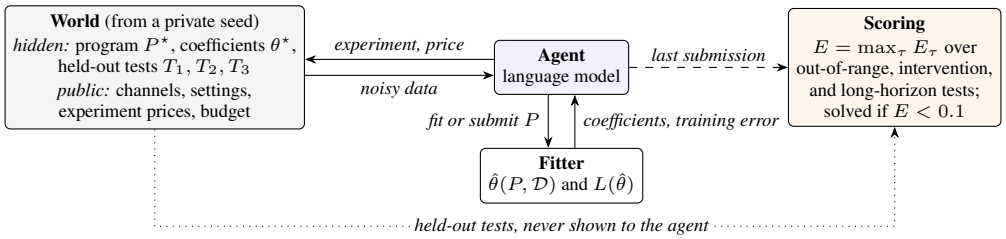


Figure 2: An episode. The agent buys experiments, the fitter estimates the coefficients of its programs, and its last submission is scored on held-out tests it never sees.

UNSATZ provides these properties as follows. It generates each system, which we call a *world*, from a private seed as a program in a small modeling language, and LLM agents write their hypotheses in the same language (i). One fitter estimates the coefficients of every proposed program, and a world is accepted only if its true program, fitted to affordable experiments, passes the tests (ii). The final model is scored only on held-out experiments: conditions outside the ranges the agent could explore, interventions it could not perform, and runs several times longer than a standard run (iii). Experiments are priced, and each budget is a small multiple of the cost of a set of experiments from which the true equations can be fitted (iv).

Discovery also accumulates across systems. The same saturating law describes enzyme kinetics, nutrient uptake by microbes, and the feeding of predators, and a scientist who has found a mechanism in one system tries it in the next, but must drop it where it does not act. UNSATZ therefore also contains *transfer problems*, which embed one generated mechanism in two different systems. After an agent investigates the first, a fresh instance of it investigates the second, starting from the program the first investigation produced. Two comparisons show whether a gain comes from the mechanism itself: a fresh agent on the second system without the program, and a control system identical to the second except for the mechanism, where the program should not help.

We evaluate nine language models on 80 worlds from 20 domains and on four transfer problems (36 transfer studies). GPT 6 Astra solves 48 of the 80 worlds and Claude Fable 5.1 solves 38, each significantly more than the next agent on a paired test; the remaining seven solve between 17 and 24 and are not separable from one another (Table 3). On the 44 worlds where we also run classical baselines, these methods solve 3 worlds in 76 attempts, whereas the fitter, given the true equations with unknown coefficients and the same data, solves 30 of the 44. Latent states are a major source of difficulty in the dynamical worlds: excluding the networks, agents solve worlds without them in 71% of episodes and worlds with them in 23%. Stronger agents make fewer unscientific errors, such as final programs that ignore the experimental settings, but for every agent most failures are scientific: agents find the hidden states and the interaction graph of a system but not the equations that govern them, and select revisions by training error rather than by prediction. Giving an agent a mechanism discovered on one system helped it on another only when the two shared the mechanism and the agent had solved the first system (5 of 6 such cases, against 2 of 6 for agents starting fresh).

2 THE UNSATZ BENCHMARK

An *episode* is one agent's investigation of one world (Figure 2): it reads the world's description, buys experiments, receives fitted coefficients for the programs it proposes, and commits a final program, scored afterwards on tests it never sees.

2.1 WORLDS

A world consists of a hidden program $P^\star$ with coefficients $\theta^\star$, a public description, and hidden test experiments. We call $P^\star$ without its coefficient values the world's *true structure*: the correct ansatz, which leaves only $\theta^\star$ to be estimated. Most worlds are dynamical systems in state-space form

$$\dot{x} = f(x, z, u; \theta), \qquad \dot{z} = g(x, z, u; \theta), \qquad y = h(x, z; \theta) + \varepsilon, \tag{1}$$

where the dot is the derivative with respect to time, x are states that experiments can prepare or perturb, z are *latent* states that are never measured directly, u are inputs set by the experimenter, and

θ are coefficients. The *observation map* h turns the states into the measured signals, or *channels*, y, which carry Gaussian sensor noise ε. Other worlds replace the differential equations by discrete-time maps or by delay or stochastic differential equations (SDEs), make x and z fields over one spatial dimension whose dynamics f and g contain spatial derivatives (partial differential equations), or are static causal and input–output models, or jump, agent-based, and lattice processes (Table 2). If f, g, and h were known, the task would reduce to estimating θ; the agent must instead find these functions, including the latent states they involve.

The description, which forms the agent's system message (Figure 1c; Appendix L), contains a two-sentence narrative of the setting, the channels and inputs with their settable ranges, the standard initial condition and run length, the experiment types with their prices, the budget, the grammar of the modeling language, and the scoring rule. It states that every channel is noisy and that some may be uninformative: in nine domains, a world can contain up to two unlabeled auxiliary channels carrying only noise or a transformed or delayed copy of a real signal. It does not reveal the equations, the coefficients, the latent states or their number, the noise level, or which channels are auxiliary.

2.2 Experiments

Each experiment is a fresh simulation of the hidden system and returns noisy measurements of every channel at fixed times. A dynamical world offers up to seven types (Appendix C): (1) a *standard run* from the standard initial condition, (2) a run from initial values of the measured states chosen within an ordinary range, (3) a run at a chosen value of an input, (4) a run in which one measured state receives an *impulse* of chosen size at a chosen time, (5) a run in which one measured state is *clamped* at a chosen value over a chosen interval, (6) a *probe*, which may start from a wider range of initial values and last up to three times as long as a standard run, and (7) an *assay*, which reads the exact value of one latent state at one time of the most recent run. Each type answers a different question. A clamp cuts the feedback through the held state, so the response of the others shows how that state acts on them, and an impulse shows how a perturbation spreads and decays. Probes and input changes expose nonlinearities and slow processes that the ordinary range hides, such as saturation or the depletion of a resource. Assays show whether a latent state exists and, repeated at several times, how it evolves; a request for a latent state beyond the last one returns nothing, so even the number of latent states can be learned, at a price. Assay values guide the agent's reasoning but do not enter the fit. Static worlds answer queries at chosen inputs, causal worlds return samples of all variables with or without one of them clamped, and stochastic worlds return ensembles of independent paths. Each experiment has a price in budget units that depends on its type and, for samples or paths, on their number and duration. The budget caps the total price of the experiments an agent buys, and fits and submissions are free.

2.3 The modeling language

The agent states each hypothesis as a program in the modeling language that also defines the worlds. A program names its class, declares its states, which of them are latent, its inputs, and the coefficients to be fitted, and gives one evolution equation per state and one observation equation per channel it predicts. For instance, the program

```
class ODE        param alpha, beta, gamma
state a, r       d a/dt = alpha*a*r - beta*a
latent r         d r/dt = u - gamma*a*r
input u          obs y1 = a
```

proposes that a measured population a grows by consuming a latent resource r, which is replenished at the feed rate u; `obs y1 = a` states that channel `y1` reads a directly, so that an impulse or a clamp applied to `y1` acts on a. Expressions combine arithmetic with a fixed vocabulary of responses, namely signed powers, saturations, exponentials, logarithms, hyperbolic tangents, sines, and threshold switches, and some classes add delays, spatial derivatives, event rates, or interaction energies (Appendix D). A program may have at most 16 fitted coefficients, including the initial value of each latent state, which no experiment can set. The language fixes this vocabulary but not the model: the number of states, which of them are latent, how they interact, and how they are measured are the agent's decisions. Because every world is itself a program in this language, a correct ansatz is

always expressible, and because programs are executable, the scorer can simulate any hypothesis under experiments that were never run.

2.4 THE FITTER

The agent never sets coefficients itself. When it fits or submits a program P, the fitter simulates P under every experiment e in the purchased dataset $\mathcal{D}$ and chooses the coefficients θ that minimize the training error

$$L(\theta) = \frac{1}{N} \sum_{e \in \mathcal{D}} \sum_{c,t} \mathrm{clip}_{[-10,10]} \left(\frac{\hat{y}_c(t; P, \theta, e) - y_c(t; e)}{s_c} \right)^2, \tag{2}$$

where c ranges over the channels that P predicts, s_c is the scale of channel c in $\mathcal{D}$ (its standard deviation plus 5% of its mean absolute value), and N is the number of residuals. The scaling makes channels of different magnitude count equally, and the clipping keeps one diverging simulation from dominating the fit. The minimizer is trust-region least squares (Virtanen et al., 2020) from six random starts of at most 60 residual evaluations each, one of which the agent may replace by coefficients of its choice. The agent receives the fitted coefficients, the training error, and the error on each experiment. Programs with random output are fitted to the distribution of the data instead (Appendix C). The agent acts through three tools, one that performs an action (an experiment, a fit, or a submission), one that re-reads earlier responses, and one that ends the episode, and it cannot run code of its own. The fitter's output is therefore its only quantitative feedback on a hypothesis, and the coefficients of every agent and every baseline are estimated in the same way, so a failure reflects the structure or the data rather than the estimation (ii). Section 4.1 confirms that a much larger fitting budget does not change the outcome.

2.5 SCORING

Only the last submitted program is scored, after the episode, on hidden test experiments in three groups: (T_1) runs from initial conditions or inputs outside the ordinary ranges, (T_2) runs under interventions and altered conditions, including types and magnitudes of intervention that the agent could not buy, and (T_3) runs several times longer than a standard run, ten times in the ODE and map worlds. These tests separate a mechanism from a curve fit. A wrong structure can fit the purchased data closely, whether by overfitting them or because it agrees with the truth only within the explored range, yet fail to predict how the system behaves far from those data, under an unfamiliar intervention, or over a much longer time (Figure 3). For each group τ,

$$E_\tau = \frac{1}{|T_\tau|\,|C|} \sum_{e \in T_\tau} \sum_{j \in C} \frac{\mathrm{RMSE}_t\big(\hat{y}_j(t; e),\, y_j^\star(t; e)\big)}{n_{\tau j}}, \qquad E = \max_\tau E_\tau, \tag{3}$$

averages, over the group's experiments and the real channels C, the root-mean-square error of the prediction $\hat{y}_j$ against the noiseless true trajectory $y_j^\star$, divided by the spread $n_{\tau j}$ of that channel over the group (its standard deviation plus 5% of its mean absolute value). The world's error E is the worst of the three groups, and a deterministic world is *solved* when $E < 0.1$: even on its hardest group of tests, the model's error is below a tenth of the system's own variation. A program that diverges, omits a real channel, or is never submitted fails. Stochastic worlds are scored on distributions: at checkpoints along each test run, the continuous ranked probability score (CRPS) of each predicted channel may exceed that of the true distribution only by a small margin (Gneiting & Raftery, 2007), and in SDE worlds the predicted probability of crossing a fixed barrier must be close to the true one (Brier, 1950) (Appendix G). Because the score compares behavior rather than formulas, any program equivalent to the truth passes.

2.6 WORLD GENERATION

A *domain* is a generator of worlds of one kind; each seed draws a new program from it, so the worlds of a domain share a kind of system but not their equations. Table 2 lists the 20 domains, grouped by the kind of system. In five of them (mechanics, kinetics, discrete dynamics, causal systems, and scaling relations), the generator draws a random interaction graph and writes each equation as a sum of one to four products of responses from the language's vocabulary, applied to the variables that feed it, with log-uniform coefficient magnitudes and random signs; up to two states are latent, and the

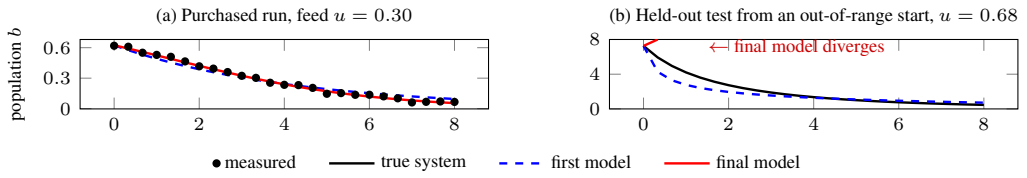


Figure 3: Two submissions of DeepSeek on an ecology world, showing the second population. (a) On a purchased run at the lowest feed rate, both fit the data, and over all eight purchased runs the final model's training error is a quarter of the first's. (b) On a held-out test starting both populations near 7.3, beyond the range the agent could prepare, the final model diverges.

Table 2: Domains by kind of system. ODE, SDE, DDE, and PDE denote ordinary, stochastic, delay, and partial differential equations; S1 and S2 are the 12- and 32-node network domains. † marks the eleven domains on which the classical and known-structure baselines are also run (Section 3).

Domain	Class	What must be inferred	Domain	Class	What must be inferred
Deterministic dynamics			*Static and causal systems*		
Mechanics†	ODE	Forces, driving, and couplings	Causal†	Causal	Dependencies under intervention
Kinetics†	ODE	Reactions with hidden species	Scaling†	Static	Relations via intermediate quantities
Discrete†	Map	Nonlinear update rules	Metrology†	Static	A latent response, two sensors
Ecology†	ODE	Two hidden resource stores	Spectroscopy†	Static	Peaks under four settings
Network†, S1†, S2	ODE	Directed couplings of nodes	*Stochastic systems*		
Reservoir	ODE	Exchanges with hidden stores	Stochastic dyn.†	SDE	Drift, noise, barrier crossing
Delay	DDE	Delayed feedback and its lag	Jump	Jump	Event rates and state changes
Transport	PDE	Fluxes, diffusion, losses	Agents	Agent rule	Update rules on a graph
Reaction–diffusion	PDE	Reactions and diffusivities	Lattice	Spin lattice	Local interaction energies
Wave	PDE	Restoring and damping forces			

sensors may mix or saturate states. The other domains instantiate templates of their own, such as the consumer–resource dynamics of the ecology world or random directed graphs of 5, 12, or 32 nodes for the networks, whose response functions are again drawn from the vocabulary (Appendices B and E). In the three spatial domains, the agent also chooses each run's domain length, boundaries, initial and forcing profiles, and sensor positions. A candidate world is discarded if its simulations are numerically unstable or its channels barely vary.

Budgets are calibrated against the true program, so that a good investigator can succeed but must still choose. For each candidate world, a procedure that knows the true program buys, at the world's prices, designs of 8 to 60 experiments that mix standard runs, probes, input changes, and interventions, and fits the true program to each design from the true coefficients. The cheapest design on which every fit reaches $E \leq 0.08$, a margin below the pass threshold, defines the world's reference cost, and the agent's budget is that cost times a factor drawn uniformly from $[2.5, 4]$. The world is accepted only if the true program, fitted to a design that spends the whole budget, again reaches $E \leq 0.08$ on five further noise seeds. A budget thus pays for two and a half to four sufficient designs, which leaves room for exploration and for a wrong first hypothesis, while every experiment still competes with the others. Every accepted world is therefore solvable within its budget by an investigator who finds its true structure and starts the fit near the right coefficients; how hard the fit is from random coefficients is measured separately (Section 3).

2.7 Transfer problems

A transfer problem has three ODE worlds. The generator draws an expression subtree, the *mechanism*, and embeds it in a *first world* and in a *related world*, where it acts on different variables within different equations; a *control world* is identical to the related one except that the mechanism is replaced. In a transfer study, an agent investigates the first world, and the related and control worlds are each investigated twice, by a fresh instance of the agent given the first investigation's final program as a starting hypothesis and by one that starts without it, with equal budget and noise seed.

3 EXPERIMENTAL SETUP

Worlds. For every domain we use the first four worlds that pass generation and calibration, from private seed streams declared before any agent ran. All agents see the same 80 worlds and four transfer problems with the same budgets and measurement-noise seeds, so they differ only in the experiments they choose and the programs they propose.

Agents. We evaluate GPT 6 Astra, Claude Fable 5.1, Muse Spark 1.3, GLM 5.3, Grok 4.6, Kimi K3, Qwen 3.8 Max, Gemini 3.8 Flash, and DeepSeek v4.1 Flash, each at its highest supported reasoning effort with default sampling parameters (Appendix F). The whole conversation stays in context, and an episode ends when the agent finishes or reaches its context or output limit. Each agent runs one episode per world and up to five per transfer problem, 896 episodes in total.

Baselines. The eleven domains marked in Table 2 contribute 44 worlds, 40 deterministic and 4 SDE. Classical methods cannot choose experiments, so on these worlds they share a random design that buys affordable experiment types with uniformly drawn settings until the budget is spent. Three methods run on the deterministic worlds, each where it applies: (1) SINDy fits a degree-three polynomial library of the measured channels and inputs on the 24 ODE and map worlds, (2) PySR, an evolutionary symbolic-regression tool, searches formulas on the 12 static-function worlds, and (3) our own grammar search, which mutates complete programs in the modeling language, latent states included, and fits 512 candidates with the fitter, runs on all 40, for 76 method–world pairs in total (Appendix I). The *known-structure baseline* gives the fitter the true structure and the same data, starting from random coefficients. It is a handicapped reference, not an upper bound: unlike the agents, it can choose neither its experiments nor its starting coefficients.

4 RESULTS

4.1 THE STRUCTURE IS THE BOTTLENECK

Given the true structure, the fitter solves 27 of the 40 deterministic baseline worlds and 3 of the 4 SDE worlds (Appendix I). On the same data, SINDy solves none of its 24 worlds, grammar search none of its 40, and PySR 3 of its 12: 3 solved worlds in 76 method–world pairs. The data therefore suffice to solve most of these worlds once the structure is known, yet structure search almost never finds such a structure. The 13 worlds that the true structure fails to solve are mostly local minima of a fit from random starting coefficients, since from the true coefficients the fitter passes all 40 (Appendix I). Agents solve 6 of these 13 worlds, in 34 of 117 episodes, more often than the 27 that the baseline solves (44 of 243); the other seven, all with latent states, no agent solves. Nor does the fitter's budget explain the agents' failures: refitting their 248 executable but failing final models with 24 starts of 200 evaluations turns none into a pass.

4.2 STRONGER AGENTS SOLVE MORE WORLDS

Table 3 summarizes the 720 episodes. GPT 6 Astra solves 48 of the 80 worlds and Claude Fable 5.1 solves 38; no other agent solves more than 24, and the top-two ranking holds on paired tests: GPT solves 12 worlds that Claude does not and Claude 2 that GPT does not (exact McNemar test, $p = 0.013$), Claude outperforms Muse 16 to 2 ($p = 0.001$), and no pair among the other seven differs significantly (smallest $p = 0.09$). At least one agent solves 54 of the 80 worlds. Spending does not separate the agents: DeepSeek and Kimi use 90% of their budgets on average, nearly as much as GPT (93%), yet solve about a third as many worlds (Appendix M).

4.3 WEAKER AGENTS MAKE MORE UNSCIENTIFIC ERRORS

Unscientific errors, mistakes that have nothing to do with the system under study, explain only part of the gap. GPT and Claude, the top two, end 5 of their 160 episodes with a model that is missing, omits a measured channel, or fails to simulate; Gemini and Kimi, the bottom two, end 20 of 160 this way. The top pair solves 86 worlds and the bottom pair 34, so the 15 extra unscientific errors could close at most 15 of that 52-world gap; for every agent, most failures are executable models that predict poorly (Figure 4). In one spectroscopy world, Claude, Kimi, and DeepSeek all begin with programs that

Table 3: Results of the nine agents on the 80 worlds. A world is solved when its final model passes every held-out test group; the interval is a 95% Wilson score interval. "Median" is the median worst-group error E and "Invalid" counts divergent, missing, or incomplete predictions (entered as $+\infty$ in the median), both over the 64 deterministic worlds. "Truncated" counts episodes that ended at the model's context or output limit. API costs are in Appendix M.

			Solved by kind of system						
Model	Solved	95% CI	Dyn. (36)	Spatial (12)	Static (16)	Stoch. (16)	Median	Invalid	Truncated
GPT 6 Astra	48/80	49–70%	18	12	9	9	0.045	2	0
Claude Fable 5.1	38/80	37–58%	10	12	9	7	0.106	3	1
Muse Spark 1.3	24/80	21–41%	4	12	7	1	0.202	5	7
Grok 4.6	23/80	20–39%	5	9	6	3	0.301	3	3
Qwen 3.8 Max	23/80	20–39%	8	10	3	2	0.276	9	40
GLM 5.3	22/80	19–38%	7	8	6	1	0.265	9	20
DeepSeek v4.1 Flash	18/80	15–33%	4	9	0	5	0.455	10	3
Gemini 3.8 Flash	17/80	14–31%	3	7	6	1	0.348	7	0
Kimi K3	17/80	14–31%	4	9	3	1	0.425	7	12

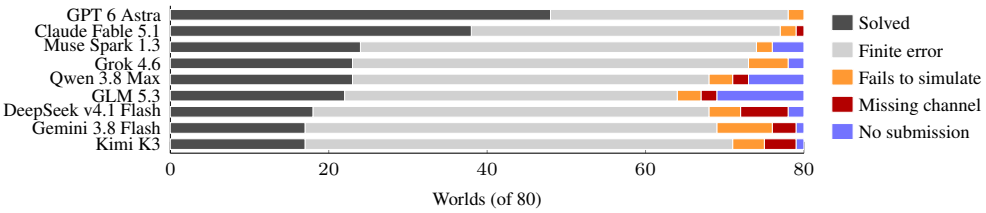


Figure 4: Outcomes of the final models; "finite error" marks executable models that predict poorly.

do not bind the four experimental settings to program variables (declarations such as `obs field = fi`), so their predictions ignore the settings. Claude tests a binding on a minimal example and fixes its program (held-out error 0.016), whereas Kimi attributes the failure to the fitter, DeepSeek substitutes a hand-derived formula, and both submit constants they declare validated.

4.4 Agents find the variables but not the mechanisms

Performance varies strongly by domain (Appendix M), and latent states explain much of the spread over the 48 deterministic dynamical worlds. Excluding the networks, agents solve the worlds without a latent state in 71% of episodes (96 of 135) and the worlds with at least one in 23% (43 of 189), falling to 12% (14 of 117) with two or three. The networks, whose nodes are all measured, are hard for another reason: GPT solves 8 of their 12 worlds, the other eight agents together 4 of 96.

The investigation records (Appendices J and K) locate the scientific errors: agents find which variables a model needs and which interact, but not how they act, and the training error hides this.

Hidden states are found but not modeled. On the 18 hidden-state worlds that offer assays, agents buy an assay in 122 of 162 episodes and, in 76, ask for a latent state beyond the last one, which reveals how many exist; 69 final models declare the true number. Yet only 19 of the 162 episodes solve the world, and final models with the true number of latent states succeed no more often (8 of 69) than the others (11 of 93). Detecting hidden states is the easy part of the ansatz; writing down how they evolve is the hard part. GPT's successful ecology episode shows what this requires: once its assays establish that the world has exactly two latent states, it fits one- and two-resource models side by side as experiments accumulate and submits one that predicts the truth (Figure 1b) within tolerance ($E = 0.091$), although the two differ in which population depletes which resource.

The right graph with the wrong mechanism. The same gap appears where every variable is measured. In a twelve-node network world, GPT and DeepSeek both recover all sixteen directed edges of the true graph. GPT submits nonlinear node dynamics with saturating couplings $\tanh(x_j - x_i)$ and reaches $E = 0.094$, DeepSeek keeps linear dynamics and couplings and reaches 0.97. Recovering the interaction graph, the goal of causal discovery, is not enough to predict the system.

Every channel is treated as signal. The prompt warns that some channels may be uninformative, yet 115 of the 127 final models on worlds with auxiliary channels predict at least one of them. Modeling such a channel costs coefficients and latent states that the real mechanism then lacks: every latent state in DeepSeek's four ecology programs (Figure 3) reproduces the third channel, a copy of the second population delayed by two sampling intervals, and none represents a resource. Deciding what not to model is part of an ansatz, and agents rarely make it.

The training error misleads revisions. DeepSeek's final ecology model, fitted to the same eight runs as its first, lowers the training error by 76%, but its held-out error rises from 0.29 to 126 (Figure 3). The revision replaces the stabilizing term $-0.054\,b^3$ of the second population b by a feed-dependent quadratic, so that at the lowest feed rate $\dot{b} = -0.427\,b + 0.441\,b^2$ grows without bound once $b > 0.97$, which the purchased low-feed runs never reach. Selecting by training error, the only signal agents see, rarely turns a solved world into a failure (7 of the 720 episodes), but in 235 of the 510 episodes with more than one submission the final model predicts worse than an earlier one. On the 44 baseline worlds, where every submission's held-out error is known, the later model had the lower training error in 35 of 48 regressions between models fitted to the same data.

4.5 Do agents reuse a discovered mechanism?

Table 10 (Appendix E) reports the 36 studies. In one, the first investigation ended without a submitted program. In the other 35, agents given the earlier program solve the related world in 6 studies and agents starting fresh in 4; on the control world, which lacks the mechanism, the counts are 0 and 1. The difference comes from the 6 studies whose first world was solved: there, agents given the program solve the related world in 5 and agents starting fresh in 2, and neither solves the control world. A discovered mechanism helps where it acts, and only if it was discovered correctly.

5 Related Work

Equation discovery. Bongard and Lipson reverse-engineered dynamical systems by choosing experiments, and Schmidt and Lipson distilled free-form laws from motion data (Bongard & Lipson, 2007; Schmidt & Lipson, 2009). SINDy selects equations from a library of candidate terms (Brunton et al., 2016), with extensions to spatial derivatives, learned coordinates, and delay embeddings (Rudy et al., 2017; Champion et al., 2019; Bakarji et al., 2023). AI Feynman and PySR advance symbolic regression, SRBench and LLM-SRBench benchmark it, and LLM-SR and BoxLM bring language models to equation and program search (Udrescu & Tegmark, 2020; Cranmer, 2023; La Cava et al., 2021; Shojaee et al., 2025a;b; Li et al., 2024). All start from fixed data with chosen variables.

Interactive scientific discovery. Bayesian experimental design chooses experiments for a given model family (Huan & Marzouk, 2013). Table 1 compares UNSATZ with the closest benchmarks (Jansen et al., 2024; Gandhi et al., 2025; Zheng et al., 2026; Chen et al., 2025a; Koblischke et al., 2025; Wiemann et al., 2026), and COAT proposes causal variables from unstructured data (Liu et al., 2024). Like LiveBench, FrontierMath, and DyVal (White et al., 2025; Glazer et al., 2024; Zhu et al., 2024), UNSATZ counters contamination (Deng et al., 2024) with fresh items from private seeds.

6 Conclusion

Unscientific errors are rare for the strongest agents, but scientific errors remain for all: agents that recover a system's hidden states or interaction graph still write inadequate equations for them and accept revisions that lower the training error. Both kinds would be caught by one habit, testing a candidate where it could fail rather than where it was fitted. We introduced UNSATZ, a benchmark in which agents must construct the structure of a scientific model, and not only its coefficients, by experimenting on generated systems. The strongest of nine language models solves 48 of 80 worlds.

AI USE STATEMENT

Generative AI tools assisted with software implementation, experiment orchestration, literature search, illustration, and manuscript preparation. The authors are responsible for the content, experimental claims, and accompanying software.

ETHICS STATEMENT

The benchmark uses simulated systems without human-subject or personal data. Model evaluation consumes commercial inference resources. The intended use is scientific capability measurement and methodological research.

REPRODUCIBILITY STATEMENT

The evaluation is determined by the generator, world and measurement seeds, experiment records, submitted programs, fitted parameters, and scorer. These records are retained for all reported episodes, and the generator, the evaluation code, the seeds of the evaluated worlds, the prompts, and the records of all 896 episodes will be released. Section 2 and Table 4 specify the protocol, and the appendices give per-setting generators, tools, the modeling language, model configurations, and the submitted programs discussed in the text. Replaying a recorded scientific evaluation does not depend on reproducing a proprietary endpoint's future responses.

REFERENCES

Joseph Bakarji, Kathleen Champion, J. Nathan Kutz, and Steven L. Brunton. Discovering governing equations from partial measurements with deep delay autoencoders. *Proceedings of the Royal Society A*, 479(2276):20230422, 2023. doi: 10.1098/rspa.2023.0422.

Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023. doi: 10.1038/s41586-023-06792-0. URL https://doi.org/10.1038/s41586-023-06792-0.

Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007. doi: 10.1073/pnas.0609476104. URL https://doi.org/10.1073/pnas.0609476104.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950. doi: 10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2. URL https://doi.org/10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016. doi: 10.1073/pnas.1517384113. URL https://doi.org/10.1073/pnas.1517384113.

Kathleen Champion, Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019. doi: 10.1073/pnas.1906995116. URL https://doi.org/10.1073/pnas.1906995116.

Yimeng Chen, Piotr Piękos, Mateusz Ostaszewski, Firas Laakom, and Jürgen Schmidhuber. PhysGym: Benchmarking LLMs in interactive physics discovery with controlled priors. In *Advances in Neural Information Processing Systems*, volume 38, pp. 52179–52217, 2025a. doi: 10.52202/085713-1556. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/42abcf9dffa48ca44fea1c497539a914-Abstract-Datasets_and_Benchmarks_Track.html.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *The*

Thirteenth International Conference on Learning Representations, 2025b. URL https://arxiv.org/abs/2410.05080.

Miles Cranmer. Interpretable machine learning for science with PySR and SymbolicRegression.jl, 2023.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. Investigating data contamination in modern benchmarks for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8706–8719. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.482. URL https://aclanthology.org/2024.naacl-long.482/.

Kanishk Gandhi, Michael Y. Li, Lyle Goodyear, Agam Bhatia, Louise Li, Aditi Bhaskar, Mohammed Zaman, and Noah D. Goodman. BoxingGym: Benchmarking progress in automated experimental design and model discovery. *arXiv preprint arXiv:2501.01540*, 2025. URL https://arxiv.org/abs/2501.01540.

Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, Olli Järviniemi, Matthew Barnett, Robert Sandler, Matej Vrzala, Jaime Sevilla, Qiuyu Ren, Elizabeth Pratt, Lionel Levine, Grant Barkley, Natalie Stewart, Bogdan Grechuk, Tetiana Grechuk, Shreepranav Varma Enugandla, and Mark Wildon. FrontierMath: A benchmark for evaluating advanced mathematical reasoning in AI. *arXiv preprint arXiv:2411.04872*, 2024. URL https://arxiv.org/abs/2411.04872.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Xun Huan and Youssef M. Marzouk. Simulation-based optimal Bayesian experimental design for nonlinear systems. *Journal of Computational Physics*, 232(1):288–317, 2013. doi: 10.1016/j.jcp.2012.08.013.

Peter Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. DiscoveryWorld: A virtual environment for developing and evaluating automated scientific discovery agents. In *Advances in Neural Information Processing Systems*, volume 37, pp. 10088–10116, 2024. doi: 10.52202/079017-0324. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/13836f251823945316ae067350a5c366-Abstract-Datasets_and_Benchmarks_Track.html.

Nolan Koblischke, Hyunseok Jang, Kristen Menou, and Mohamad Ali-Dib. Gravity-Bench-v1: A benchmark on gravitational physics discovery for agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 31084–31104. PMLR, 2025. URL https://proceedings.mlr.press/v267/koblischke25a.html.

William La Cava, Patryk Orzechowski, Bogdan Burlacu, Fabrício Olivetti de França, Marco Virgolin, Ying Jin, Michael Kommenda, and Jason H. Moore. Contemporary symbolic regression methods and their relative performance. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://arxiv.org/abs/2107.14351.

Michael Y. Li, Emily Fox, and Noah D. Goodman. Automated statistical model discovery with language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 27791–27807. PMLR, 2024. URL https://proceedings.mlr.press/v235/li24v.html.

Chenxi Liu, Yongqiang Chen, Tongliang Liu, Mingming Gong, James Cheng, Bo Han, and Kun Zhang. Discovery of the hidden world with large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 102307–102365, 2024. doi: 10.52202/079017-3249.

URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/b99a07486702417d3b1bd64ec2cf74ad-Abstract-Conference.html.

Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall, Upper Saddle River, NJ, 2 edition, 1999.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024. doi: 10.48550/arXiv.2408.06292. URL https://arxiv.org/abs/2408.06292.

Samuel H. Rudy, Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017. doi: 10.1126/sciadv.1602614. URL https://doi.org/10.1126/sciadv.1602614.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009. doi: 10.1126/science.1165893. URL https://doi.org/10.1126/science.1165893.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K. Reddy. LLM-SR: Scientific equation discovery via programming with large language models. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=m2nmp8P5in.

Parshin Shojaee, Ngoc-Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa D. Doan, and Chandan K. Reddy. LLM-SRBench: A new benchmark for scientific equation discovery with large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55325–55359. PMLR, 2025b. URL https://proceedings.mlr.press/v267/shojaee25a.html.

Silviu-Marian Udrescu and Max Tegmark. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020. doi: 10.1126/sciadv.aay2631. URL https://doi.org/10.1126/sciadv.aay2631.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, 2020. doi: 10.1038/s41592-019-0686-2. URL https://doi.org/10.1038/s41592-019-0686-2.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Venkat Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. LiveBench: A challenging, contamination-limited LLM benchmark. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2406.19314.

Matt L. Wiemann, Lindsay M. Smith, Peter Melchior, Siddharth Mishra-Sharma, Andrew Gordon Wilson, Pavel Izmailov, and Carolina Cuesta-Lázaro. DiscoverPhysics: Benchmarking LLMs for out-of-the-box scientific thinking. *arXiv preprint arXiv:2605.26087*, 2026. URL https://arxiv.org/abs/2605.26087.

Tianshi Zheng, Kelvin Kiu-Wai Tam, Newt Hue-Nam K. Nguyen, Baixuan Xu, Zhaowei Wang, Jiayang Cheng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Tianqing Fang, Yangqiu Song, Ginny Y. Wong, and Simon See. NewtonBench: Benchmarking generalizable scientific law discovery in LLM agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.07172.

Kaijie Zhu, Jiaao Chen, Jindong Wang, Neil Zhenqiang Gong, Diyi Yang, and Xing Xie. DyVal: Dynamic evaluation of large language models for reasoning tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.17167.

A PROTOCOL CONFIGURATION

Table 4 collects the constants of the protocol described in Section 2.

Table 4: Protocol configuration. Appendices B–E give per-domain details.

Component	Configuration
Programs	At most 16 fitted parameters, including latent initial values; at most 400 expression nodes (1,200 for Network-S2)
Generators	1–4 terms per equation and 1–2 factors per term; power exponents $\{-1, -\frac{1}{2}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, 1, \frac{3}{2}, 2, 3\}$; up to 2 latent states; 0–2 auxiliary channels; signal-to-noise ratio log-uniform in $[30, 1000]$ in mechanics, kinetics, discrete, causal, and scaling
Fitter	Trust-region reflective least squares, 6 starts of at most 60 residual evaluations each (finite-difference Jacobian columns are not counted); starting coefficients have random sign and log-uniform magnitude in $[0.03, 8]$, latent initial values are uniform in $[-2, 2]$, and an agent-supplied start replaces one of the six; stochastic processes use Nelder–Mead, 6 starts of 60 evaluations
Held-out tests	$(\lvert T_1\rvert, \lvert T_2\rvert, \lvert T_3\rvert)$: $(8, 6, 3)$ for ODE and map worlds, $(4, 3, 2)$ for delay and spatial worlds, $(3, 1, 1)$ for jump, agent, and lattice worlds; causal and scaling 10 and 6; metrology and spectroscopy 16 or 24 and 8; SDE 4 conditions; stochastic scores use 512 simulated paths per condition
Stochastic scores	Excess CRPS $\mathbb{E}\lvert X - Y\rvert - \frac{1}{2}\mathbb{E}\lvert X - X'\rvert - \frac{1}{2}\mathbb{E}\lvert Y - Y'\rvert$ for predicted X, X' and true Y, Y', divided by the true standard deviation (at least 0.1) and maximized over recorded times; excess Brier score $(\hat{p} - p)^2$
Calibration	True-coefficient fits on designs of 8, 12, 16, 24, 32, 44, or 60 experiments on three noise seeds; five further seeds at the full budget
Long-horizon tests	Ten times the standard run in the ODE and map domains, four times in the delay and spatial domains, and three times in the jump, agent, and lattice domains

B SCENARIO SPECIFICATIONS

This appendix specifies the mechanics, kinetics, discrete, causal, scaling, ecology, network, Network-S1, metrology, spectroscopy, and stochastic-dynamics domains, the domains marked † in Table 2; Appendix E specifies the others. The specifications below describe their generative distributions and experimental interfaces. Each evaluated agent receives a domain narrative, channel names, standard preparation, controllable ranges, action prices, acquisition budget, and the common hypothesis language. The sampled equations, parameters, latent variables, sensor assignments, and hidden test conditions remain withheld during discovery. Table 5 gives the public numerical configuration of the 44 evaluated worlds; world numbering agrees with the per-world result tables.

Table 5: Public configuration of the 44 worlds in these domains. Channels include any auxiliary signals. T is the standard simulated duration, measured in ticks for discrete maps; extended runs use longer grids. A stochastic run returns an ensemble with the indicated default number of samples per path. Budgets are in budget units, distinct from API expenditure.

Setting	Channels	T	Samples/run	B_1	B_2	B_3	B_4
Mechanics	2–4	1.46–16.9	41	337	89	179	187
Kinetics	2–3	0.655–12.2	41	120	178	322	102
Discrete	3–5	30	31	121	246	203	181
Causal	5–6	—	Point/sample	160	2,135	171	1,181
Scaling	4–7	—	Point/sample	215	149	109	88
Ecology	2–3	8	25	243	166	192	144
Network	5–6	6	25	236	196	111	182
Network-S1	12	4	25	156	167	140	141
Metrology	4	—	Point/sample	134	148	179	199
Spectroscopy	5	—	Point/sample	180	77	171	138
SDE	2	1.5	31/path	5,660	6,289	2,950	5,024

B.1 COMMON CONSTRUCTION AND OBSERVATION MODEL

The mechanics, kinetics, discrete, causal, and scaling generators construct equations as sums of products of elementary responses. Before the parameter allowance is imposed, an equation draws one to four interaction terms with probabilities $(0.25, 0.45, 0.25, 0.05)$; each term contains one or two factors with probabilities $(0.65, 0.35)$. The dynamical generators reserve parameters for characteristic scales, hidden initial conditions, and optional switching behavior, then restrict the term count to fit the

resulting allowance. Coefficient magnitudes are sampled on a logarithmic scale, with signs selected independently for interaction terms. Characteristic scales are shared across terms involving the same coordinate.

The continuous-dynamics atom pool comprises signed powers, saturation, exponentials, logarithmic responses, hyperbolic tangents, oscillatory forcing, and compositions. Static equations omit explicit time dependence; the map profile emphasizes saturating and other nonlinear update responses. Power exponents are drawn from

$$\left\{-1, -\tfrac{1}{2}, \tfrac{1}{3}, \tfrac{1}{2}, \tfrac{2}{3}, 1, \tfrac{3}{2}, 2, 3\right\}, \tag{4}$$

and saturation exponents from $\{\frac{1}{2}, 1, 2, 3, 4\}$. In the mechanics, kinetics, and discrete domains, an equation additionally receives a negative self-response with probability 0.85. Composed atoms apply an exponential, saturation, logarithm, or hyperbolic tangent to a powered coordinate. The exact meanings of these operators are given in Appendix D.

Measurements combine a generating observation function with independent Gaussian sensor noise. For the mechanics, kinetics, discrete, causal, and scaling domains, the signal-to-noise ratio is sampled logarithmically between 30 and 1,000; channel noise scales are determined from reference observations with a scale floor. Zero, one, or two auxiliary channels are added with probabilities $(0.40, 0.35, 0.25)$. They can be noise-dominated signals, transformed combinations of genuine channels, or lagged channels in dynamical domains. Channel order is shuffled, and the agent receives no label identifying auxiliary signals. Behavioral scoring uses the genuine observation channels.

Each acquisition specifies a fresh experiment: an initial-condition change or input setting applies to that run, rather than permanently altering subsequent runs. A standard observation returns to the disclosed standard preparation. The most recent trajectory is retained for an optional internal assay. Acquired measurements accumulate for fitting throughout the episode.

Deterministic ODEs use fixed-step, classical fourth-order Runge–Kutta integration. Maps advance synchronously by one update per tick. Stochastic systems use the separate Euler–Maruyama configuration specified below.

For mechanics, kinetics, discrete dynamics, ecology, and both network domains, the eight OOD tests draw preparations beyond the ordinary box. A signed coordinate uses its box midpoint plus or minus three to six half-widths; a positive-preparation coordinate uses three to six times its ordinary upper bound. Half of these conditions use extended input values when an input exists. Intervention tests combine clamps, impulses, and input changes, including unavailable intervention types or values outside the ordinary action bands. Public long probes reach three times the standard duration, while hidden long tests reach ten times that duration with proportionally longer observation grids. Causal and scaling OOD root settings use the same shell construction. Metrology and spectroscopy instead draw within their extended boxes while placing at least one coordinate outside its ordinary range.

B.2 Mechanics

The scenario is an instrumented mechanical assembly with unknown effective coordinates and couplings. Its generating model is an ODE with two to four states. Up to two states are initially designated unobserved, subject to retaining at least two observed coordinates. A directed graph defines candidate interactions: edge probability is selected from $\{0.35, 0.50, 0.65\}$, and proposals require incoming connections and a path from each latent coordinate to an observed one. The equations select and compose responses from these permitted neighbors. A constant external drive u is included with probability 0.5 and acts on one or two states. Coefficient magnitudes lie in $[0.2, 2.5]$, and explicit oscillatory forcing can depend on time.

Coordinate scales lie in $[0.5, 2]$. Ordinary preparations span $[-s, s]$ and wider preparations $[-2s, 2s]$. When present, the ordinary drive range is $[0.5, 1.5]$. Observations can mix coordinates through a signed-square or multiplicative term, and a channel can have a saturating readout. Such transformations make the underlying state distinct from the measured signal. An optional thresholded response, proposed with probability 0.4, becomes active beyond the ordinary regime.

The standard horizon is chosen from the characteristic rate at the default preparation and clipped to $[0.5, 60]$. Each standard trajectory contains 41 recorded times, with four Runge–Kutta substeps between adjacent observations. Actions include ordinary runs and chosen preparations, with impulses, clamps, drive changes, and wider/longer probes offered according to the world's disclosed action list.

An internal assay is always listed. Hidden tests comprise eight OOD conditions, six interventions or altered inputs, and three runs at ten times the standard horizon. This domain tests coupled dynamics, hidden state, nonlinear readouts, and mechanisms that emerge outside short ordinary runs.

B.3 KINETICS

The scenario is a reaction vessel whose assayed signals represent deviations from a reference state. It uses the same two-to-four-state ODE construction and observation transformations as mechanics, with coefficient magnitudes in $[0.05, 5]$ and no explicitly time-dependent forcing atoms. Ordinary initial preparations lie in $[0.1s, s]$, and extended preparations in $[0.1s, 2s]$; the state variables themselves represent deviations and can evolve through signed values. An optional input represents a feed setting.

Experiments correspond to preparing species, injecting a bolus, holding a coordinate with a feed, changing the external setting, or assaying an unobserved species. The horizon, numerical integration, optional action availability, and $8/6/3$ hidden-test counts follow the continuous mechanics profile. The structural challenge is to explain jointly the observed reaction-like dynamics and the influence of unmeasured quantities, including their response to interventions and long-term depletion.

B.4 DISCRETE DYNAMICS

The scenario is a synchronous update process observed through noisy counters. Two to four states follow

$$x_{k+1} = x_k + F(x_k, u; \theta), \tag{5}$$

with independently composed nonlinear increments, optional unobserved states, transformed measurements, and an optional external rate. The update-response pool contains saturation, exponentials, logarithmic responses, hyperbolic tangents, and compositions; an additional thresholded response can introduce a regime change. Ordinary and wider preparation ranges follow the signed mechanics profile.

A standard experiment records 31 values, including the initial state and thirty updates. Setting a counter changes its initial value, an impulse adds to a counter during the run, and a clamp pins a coordinate over an interval. A long public probe extends the run to ninety updates; hidden long-horizon tests use 300 updates. Eight OOD, six intervention, and three long-horizon tests evaluate whether the inferred update rules capture the process beyond the observed sequences.

B.5 CAUSAL SYSTEMS

The scenario is a population described by jointly measured attributes. A random acyclic graph contains three to six nodes, with candidate forward-edge probability selected from $\{0.40, 0.55, 0.70\}$. Non-root nodes receive nonlinear structural equations composed from their parents. With probability 0.5, one eligible non-root, non-sink node is hidden. Root values are sampled uniformly from their ordinary boxes; non-root equations receive independent additive Gaussian structural noise with standard deviation 0.1 times the node scale, followed by sensor noise.

An observational action buys n independent joint samples. A clamp replaces one visible node's structural equation by a chosen value, then buys n samples of the intervened system. Ordinary observation costs one unit per sample, while intervention costs are disclosed per world. Hidden evaluation uses ten extended root configurations and six interventions on visible non-root nodes, comparing the noiseless structural responses. This domain requires reasoning from noisy observations and interventions to the dependencies and functional mechanisms of the system.

B.6 SCALING RELATIONS

The scenario is a measurement rig with adjustable inputs and dependent recorded quantities. It uses a three-to-six-node acyclic computation graph with the same compositional static-equation construction as the causal domain. An intermediate node can be unobserved. Here the generating relation is deterministic before sensor noise: each query specifies all root-input values, rather than sampling a population with structural noise.

Ordinary input ranges are $[0.1s, s]$ and extended ranges $[0.1s, 2s]$. An ordinary point evaluation costs one unit; wider-regime evaluations, when offered, cost more. The hidden battery contains ten OOD root configurations and six interventions on observed derived quantities. A candidate therefore specifies a complete system of relationships, including how changes to an intermediate quantity propagate, rather than only a separate regression for each output.

B.7 ECOLOGY

The scenario is a controlled microcosm with two measured populations and two hidden resource stores. The generator independently selects which resources support growth, which consumers deplete each resource, and which variables contribute feedback. Population equations combine resource-dependent growth, linear loss, and nonlinear density regulation. Resource equations combine input-driven regeneration, turnover, consumption, transfer between stores, and signed feedback. Response terms are composed from linear, saturating, hyperbolic-tangent, logarithmic, and signed-power atoms. Coefficients are shared by process role while source variables and atom choices vary across terms.

Population preparations lie in $[0.15, 1.25]$, extended preparations in $[0.05, 2.5]$, and replenishment u in $[0.3, 1.7]$. Hidden resource initial values lie in $[0.45, 1.15]$. Standard trajectories span eight time units with 25 observations and two integration substeps per observation interval. All of the dynamical acquisition types are offered, including assays of the resource stores. Zero or one auxiliary signal is added; the sensor SNR lies in $[40, 800]$.

The hidden battery uses eight extended preparations or input conditions, six interventions, and three eighty-unit trajectories. Input extrapolation includes replenishment values in $[1.9, 2.8]$. The scientific challenge is to distinguish direct population interactions from effects mediated by resource depletion, regeneration, and memory in hidden stores.

B.8 NETWORK DYNAMICS

The standard network contains five directly measured nodes. A directed graph is sampled edge by edge, with edge probability in $\{0.25, 0.35, 0.45\}$ and acceptance conditional on strong connectivity. Each node combines linear self-activation, a nonlinear restoring response, a bias, and input-scaled interactions with its incoming neighbors. Coupling functions act on differences $x_j - x_i$ and are selected from linear, hyperbolic-tangent, and signed-power responses. Restoring exponents are drawn from $\{1.5, 2, 3\}$.

Ordinary preparations lie in $[-0.35, 0.35]$, probes in $[-1, 1]$, and the global coupling input in $[0.4, 1.4]$. Standard trajectories span six time units and contain 25 observations, with two integration substeps per observation interval. Every dynamical acquisition type is offered. The five node signals can be accompanied by one auxiliary channel; measurement SNR lies in $[40, 800]$. The $8/6/3$ hidden battery includes input values in $[1.8, 2.6]$ and sixty-unit trajectories. The agent must identify both the directed graph and the nonlinear response associated with its nodes and edges.

B.9 NETWORK-S1: TWELVE-NODE SCALING SETTING

Network-S1 increases the system to twelve directly measured nodes. A randomly ordered directed cycle provides a strongly connected backbone. Additional directed edges are sampled independently with probability in $\{0.025, 0.04, 0.06\}$, retaining graphs with two to five additional edges. Backbone interactions use hyperbolic tangents; the additional interactions select from the network response atoms. Distinct shared coefficients control backbone and additional edges, and node-specific bias signs and restoring exponents create heterogeneous dynamics.

Ordinary preparations lie in $[-0.2, 0.2]$, probes in $[-0.7, 0.7]$, and the coupling input in $[0.5, 1.3]$. A standard experiment spans four time units with 25 observations and two integration substeps per observation interval. The hidden $8/6/3$ battery includes input values in $[1.5, 2.1]$ and forty-unit trajectories. This is a larger network domain under the same interaction and submission interface, with graph recovery and functional identification both contributing to performance.

B.10 METROLOGY

The scenario is a calibration rig with two controlled settings, `u1` and `u2`, and two anonymous instruments, `sensor_a` and `sensor_b`. A latent reference response is constructed from an intercept and six to eight terms. The terms include univariate powers and nonlinear responses, products of responses of the two inputs, and nonlinear functions of mixed input combinations. An outer transformation selects the direct response, a hyperbolic tangent, a linear-plus-tangent response, or a sine-modulated response.

The instruments apply different unknown transformations to the same latent response: one has saturating compression and the other a soft dead zone, with their assignment randomized. Both include an unknown gain, bias, scale, and shape exponent. Ordinary settings are in $[-1, 1]^2$ and extended settings in $[-2.5, 2.5]^2$. Channel order is shuffled. Input values are returned exactly, while instrument readings have Gaussian noise with SNR in $[100, 700]$ and standard-deviation floor 0.002.

An ordinary two-instrument measurement costs one unit and an extended measurement costs four. A `gold_measure` action costs eight and returns the noiseless latent reference at a chosen extended-range setting. This scalar can guide model construction and parameter initialization; it is separate from the sensor dataset used by the standard fitter. Hidden evaluation comprises sixteen outer-range settings and eight controlled-input changes, scoring the observable response channels. The task combines discovery of a latent relationship with separation of instrument distortion from the underlying signal.

B.11 SPECTROSCOPY

The scenario measures intensity as a function of frequency ν, field F, temperature T, and requested resolution r. A spectrum combines three or four generated peak components and a background. Each component independently selects its field-dependent center shift, resolution-dependent width, temperature broadening, thermal amplitude, and line-shape kernel. Center shifts can be linear, polynomial, sinusoidal, or saturating. Width laws combine an intrinsic scale with additive, quadratic, root-sum-square, multiplicative, or saturating resolution responses. Thermal amplitudes use activated or contrast responses. Kernels include Gaussian, super-Gaussian, Lorentzian, squared-Lorentzian, mixed Gaussian–Lorentzian, doublet, and triplet shapes. The background can depend on temperature, field, or resolution.

Table 6: Spectroscopy acquisition ranges.

Setting	Ordinary	Extended
Frequency	$[0.5, 9]$	$[0.2, 12]$
Field	$[-0.3, 0.3]$	$[-0.8, 0.8]$
Temperature	$[0.8, 1.3]$	$[0.5, 2]$
Resolution	$[0.03, 0.12]$	$[0.006, 0.18]$

Each query returns the four settings and one noisy intensity. The disclosed acquisition law is

$$c(r) = 0.25 + \frac{0.02}{r}, \qquad \sigma(r) = 0.003\sqrt{\frac{0.04}{r}}. \tag{6}$$

An ordinary query costs $c(r)$ and an extended query costs $2c(r)$. Input channels are noiseless. The hidden battery contains 24 outer-range conditions and eight controlled changes of one input. Recovering the response requires identifying how spectral components shift, broaden, and change amplitude across conditions, while allocating measurements across resolution and cost.

B.12 STOCHASTIC DYNAMICS

The scenario is a fully observed one- or two-state Itô diffusion,

$$dX_i = f_i(X, u; \theta)\, dt + g_i(X_i; \theta)\, dW_i, \tag{7}$$

with independent Wiener drivers. One and two states are proposed with probabilities 0.35 and 0.65. Each drift combines linear decay with two or three independently composed response terms,

including a driven term. Responses include linear, hyperbolic-tangent, sinusoidal, signed-logarithmic, signed-saturation, and sublinear-power atoms, and products of two atoms. A diffusion amplitude combines a positive baseline with a bounded state-dependent modulation. The generating amplitudes remain positive; the submitted language represents amplitudes directly.

Ordinary initial states and input u lie in $[-1, 1]$. Extended initial states lie in $[-2.5, 2.5]$ and extended inputs in $[-2, 2]$. Ordinary durations range from 0.25 to 1.5, and probe durations from 0.25 to 3, in increments of 0.05. The public interface specifies one to 256 independent paths per acquisition. A probe must extend at least one initial condition, input, or duration beyond its ordinary range. Cost is the path count times duration. Integration uses Euler–Maruyama with step 0.01; observations are spaced by 0.05 and include independent sensor noise with standard deviation between 0.01 and 0.02.

Hidden evaluation uses four conditions with 512 paths each. Two OOD conditions start at signed magnitudes between 2.7 and 3.6 and run for 2.5 time units. Two first-passage conditions start near a barrier and run for three time units. Each condition selects a target state channel. Distributional comparison uses normalized energy/CRPS excess at recorded-time checkpoints, and first-passage comparison uses excess Brier loss for a barrier crossing. The task requires a model of both the systematic drift and stochastic variability, rather than a fit to a particular noise realization.

C Agent tools and experimental actions

C.1 Conversation-level tools

The evaluated interface exposes exactly three tools. The full public experiment description is supplied in the initial prompt, and the action list specifies which experiments are available in that world.

`act({"action": A}).` Execute the action object A. Its `type` selects an acquisition, a fit, a submission, or hypothesis logging. The outer object contains exactly `action`; arguments such as `ic` and `program` are fields of A itself.

`response_archive({...}).` Read previously returned action responses, including complete observations and fitting diagnostics. This read-only tool also provides numerical extraction and summary statistics.

`finish({}).` End the episode. The final model is the last accepted `submit` action. The termination call itself supplies no model or fitted parameters.

Calls and responses use JSON. Experiment prices are deducted from the remaining budget; fitting, submission, archive access, and hypothesis logging are free. A rejected request returns an error without buying measurements or replacing the committed model. Successful action responses report `ok`, `cost`, `remaining`, and `experiments_run`, together with the action-specific data.

C.2 Acquisition actions

Table 7 gives the prices in these domains. A requested action must appear in the world's disclosed action list. All numerical settings must lie within the corresponding disclosed bounds, and the complete acquisition cost must fit within the remaining budget.

Table 7: Base acquisition prices across the four worlds of each domain. Ranges summarize world-specific prices; an asterisk means an action is available in only a subset of those four worlds. Causal prices are per joint sample and are multiplied by the requested sample count. For spectroscopy, $c(r) = 0.25 + 0.02/r$; stochastic prices are PH for P paths of duration H. The three free action types are described separately.

Setting	Observe	Initial	Kick	Clamp	Input	Probe	Assay	Gold
Mechanics	1–2	2–5	4–6*	8–11*	9*	6–14	15–40	—
Kinetics	1–2	2–4	4–6*	7–11*	4–10*	16*	15–40	—
Discrete	1–2	2–4	3–5*	5–11*	6–9*	9–10*	15–40	—
Causal	1	—	—	3–8	—	—	—	—
Scaling	1	—	—	—	—	4–10	—	—
Ecology	1–2	2–4	4–6	6–9	4–10	10–14	15–40	—
Network	1–2	3–4	3–6	8–10	5–9	8–13	15–40	—
Network-S1	1–2	3–5	4–6	6–10	6–7	7–14	15–40	—
Metrology	1	—	—	—	—	4	—	8
Spectroscopy	$c(r)$	—	—	—	—	$2c(r)$	—	—
SDE	PH	—	—	—	—	PH	—	—

observe. For ODE and map worlds, `{"type":"observe"}` records a run from the standard preparation. For causal worlds, `{"type":"observe","n":8}` buys eight independent joint samples; `n` defaults to one and may be any positive integer up to 4,096 that fits the budget. For static functions, `ic` must provide every controlled input, for example `{"type":"observe","ic":{"u1":0.2,"u2":-0.4}}`. For SDE worlds, `paths` is required, with optional `ic`, `inputs`, and `horizon`; omitted settings use the standard preparation and default duration.

set_initial. The action `{"type":"set_initial","ic":{"y1":0.4}}` changes the initial value of a directly controllable channel and records a fresh dynamical run. A dictionary can specify several channels; unspecified coordinates retain their standard preparations. Hidden initial states are part of the world's preparation and remain outside these controls.

set_input. The action `{"type":"set_input","inputs":{"u":1.0}}` sets an exogenous input for one run. The input is constant during that experiment. An optional `ic` changes the preparation as well.

kick. The action `{"type":"kick","channel":"y1","time":1.0,"delta":0.5}` adds an impulse to a controllable coordinate during a fresh run. The absolute impulse magnitude must lie in the disclosed band, and its time must lie within the standard horizon. An optional `ic` changes the initial preparation.

clamp. For a dynamical world, the action specifies `channel`, `value`, `t0`, and `t1`. The coordinate is held fixed over the interval, with $0 \le t_0 < t_1 \le T$; an optional `ic` changes the preparation. For a causal world, `channel` and `value` define a do-intervention, and optional `n` specifies the number of independent joint samples. The intervened structural equation is replaced by the requested value.

Combining a custom `ic` with `kick`, dynamical `clamp`, or `set_input` adds the preparation surcharge $c_{\text{initial}} - c_{\text{observe}}$ to the listed action price.

probe_regime. For ODE and map worlds, `ic` selects a preparation in the wider box and `long:true` triples the horizon and number of observation intervals. For static functions, all controlled inputs must be supplied in `ic`, within the extended box. Spectroscopy applies its resolution-dependent price. For stochastic worlds, a probe uses `ic`, `inputs`, `horizon`, and `paths`, with at least one setting outside the ordinary acquisition range.

deep_assay. The action `{"type":"deep_assay","k":1,"t":0.0}` buys a noiseless value of the kth hidden state from the most recent dynamical trajectory. Indices are one-based, and

the requested time must lie within that trajectory. The returned value uses the nearest recorded time, which is reported with it. An index beyond the hidden-state list returns `null` and still incurs the assay price. The scalar response is available for reasoning and parameter initialization; it is separate from the standard observation dataset used by fitting.

`gold_measure`. Available in metrology, this action supplies both settings in `ic` and returns the noiseless latent reference as `reference`, together with the queried `setting`. It costs eight acquisition units. Like an internal assay, it provides additional scientific evidence without appending an ordinary sensor row to the fitting dataset.

C.3 Observation formats

ODE and map acquisitions return `experiment`, a time vector `t`, ordered `channels`, and a matrix `y` indexed by time and channel. Static-function evaluations use the same fields with a single observation row. Causal acquisitions return `channels` and `samples`, a matrix indexed by independently sampled unit and channel. Successful stochastic ensembles return `experiment`, `t`, `channels`, `paths`, and `y` indexed by path, time, and channel. The reported channel order specifies the interpretation of every array column.

An ordinary deterministic trajectory contains its initial observation and one observation after each of the disclosed intervals. Impulses and clamps are included in the returned experimental specification, so later fitting uses the conditions under which each trajectory was acquired.

C.4 Fitting, submission, and hypothesis logging

`fit`. The action `{"type":"fit","program":"<DSL text>"}` estimates parameters using all purchased fitting data. Optional `init` supplies a parameter vector and optional `bounds` supplies one lower/upper pair per parameter; the order follows the program's parameter list, including any added hidden-state initial values. The response provides the ordered `params` and a `fit` object containing `theta`, `cost`, `success`, `n_starts`, `per_sample_rmse`, and elapsed fitting time.

For deterministic classes, the fitter performs multistart nonlinear least squares on simulated observations. Channel scales come from purchased data, residuals are clipped to $[-10, 10]$, and the reported cost is their mean square. Static root inputs and imposed do-values are treated as experimental conditions rather than regression targets. The standard configuration uses six starts with up to sixty least-squares evaluations per start; a supplied initialization occupies one of those starts. A parameter-free program is evaluated directly.

For stochastic programs, fitting matches empirical drift and innovation-variance moments from the acquired path increments, correcting the variance for measurement noise. It uses six starts without an initialization and one supplied start otherwise, again with up to sixty evaluations per start. SDE fitting requires exactly one directly observed channel per state, in the disclosed channel order. Identical program/data/initialization/bounds requests use the same optimizer seed and therefore reproduce the same diagnostic.

The training error uses the channels the candidate predicts. Consequently, a low training error alone does not establish that the candidate predicts every channel required by the held-out evaluation. Observation coverage, faithful execution, and generalization are separate checks.

`submit`. The action accepts the same `program`, `init`, and `bounds` fields, performs the fitting step, and records the resulting program and parameters as the current commitment. Its public response gives acceptance and fitting diagnostics. Hidden errors are withheld. A subsequent accepted submission replaces the current commitment; a rejected submission leaves it unchanged. Calling `fit` alone does not commit a model.

`log_hypotheses`. This action takes a `hypotheses` field containing a list of program strings and records those proposed explanations at the current acquisition spend. It neither fits nor submits the programs and provides no hidden-score feedback.

C.5 COMPLETE OBSERVATION ACCESS

Responses exceeding 20,000 characters are represented in the conversation by a bounded message identifying their archive entry. The archive retains the complete response. Its four operations are:

`list`. Accepts optional `offset` and `limit`; returns entry indices, action types, and response lengths. Pages contain at most 64 entries.

`get`. Requires `index` and accepts `offset` and `limit`; returns an exact JSON-text `chunk`, its position, total length, and `next_offset`. The evaluated configuration uses pages of at most 8,000 characters.

`values`. Requires `index` and exactly one of `field` or `path`. A path is a sequence of object keys and array indices. The response reports the array shape and pages its numerical entries in row-major order, with at most 512 values per page.

`stats`. Selects a field or path in the same way and accepts an optional `axis`. It returns count, minimum, maximum, mean, and population standard deviation, either globally or along the selected axis. Result arrays are paged using `offset` and `limit`.

For example, the following calls inspect an archived trajectory without buying another experiment:

```
response_archive({"operation":"list","offset":0,"limit":8})
response_archive({"operation":"get","index":0,"offset":0,"limit":8000})
response_archive({"operation":"values","index":0,"field":"y","limit":64})
response_archive({"operation":"stats","index":0,"field":"y","axis":0})
```

For a causal response the numerical field is `samples`. For an ensemble, reducing `y` along axis zero summarizes across paths. Returned `next_offset` values permit complete traversal of a paged response.

D EXECUTABLE HYPOTHESIS LANGUAGE

D.1 PROGRAM STRUCTURE

A hypothesis is a UTF-8 program with one statement per line. Blank lines and comments beginning with # are ignored. State and parameter names are chosen by the agent; observation-channel and exogenous-input names must match the experiment's description. The principal statements are:

```
class ODE | Map | SDE | StructuralDAG | StaticFunction
state x1, x2, ...
latent z1, z2, ...
input u, ...
param a, b, k, ...
d x1/dt = <expression>       # ODE or SDE drift
next x1 = <expression>       # complete next value for a Map
diffusion x1 = <expression>  # SDE diffusion amplitude
let x2 = <expression>       # static structural equation
obs y1 = <expression>       # observed-channel prediction
```

The first line selects one of the five classes. Comma-separated declarations may span multiple declaration statements. Expressions use declared states and parameters; dynamical right-hand sides can also use disclosed exogenous inputs and the time symbol `t`. Observation maps are functions of the state and parameters. The language includes numerical constants, parentheses, unary signs, and ordinary arithmetic precedence. Fixed numerical choices such as an exponent are part of the hypothesis structure; named parameters are estimated by the fitter.

Every ODE, map, or SDE state requires an evolution equation. A map's `next` statement specifies x_{k+1} itself, so an incremental model writes `next x = x + ...`. Every SDE state additionally requires an explicit `diffusion` statement, whose value multiplies its own independent Wiener increment. Static classes require at least one root state without a `let` equation. Other static states are evaluated in their `state`-declaration order, with parents preceding their consumers; the line order of `let` statements is not a substitute for this dependency order.

Programs are limited to sixteen fitted parameters and 400 expression-tree nodes. The node count includes all evolution, diffusion, and observation expressions. All referenced state and parameter names must be declared. The parser normalizes accepted notation to program text; submissions use a stable parse–print representation. The examples below are checked to round-trip through that representation.

D.2 Initial conditions, controls, and readouts

The declaration `obs y1 = x` is a direct readout. For a non-latent dynamical state it binds preparations, impulses, and clamps on channel `y1` to state `x`. An expression such as `obs y2 = z/2` instead defines a transformed measurement. State names are internal identifiers; matching a state's spelling to a channel name alone does not create a binding.

Each dynamical state without a direct controllable readout has a fitted initial-value parameter `<state>_0`. An explicit `latent` declaration also creates this parameter. These initial values are shared across experiments and count toward the sixteen-parameter limit. Directly controlled states use the experimental preparation, including standard values for coordinates that are not changed.

For a static program, roots are states without defining equations. A direct observation declaration binds an experimental input to its corresponding root, as in `obs field = field`. Thus static settings are represented by root states and readouts; the `input` declaration represents dynamical drive signals. Supplied input and do-values are bound before evaluating dependent equations. A do-intervention fixes its target instead of evaluating the target's structural equation. Unbound roots retain their default zero value. Static intermediate states are algebraic quantities and receive no dynamical initial-value parameters.

These rules connect the scientific model to the experiment. A complete static response program therefore declares its input readouts as well as its predicted outputs. SDE fitting in the evaluated profile uses a one-to-one direct state/channel correspondence, while ODE and map programs can introduce hidden states and transformed observation functions.

D.3 Expression semantics

Let $\mathrm{spow}(x, r) = \mathrm{sign}(x)|x|^r$ denote signed power. For negative exponents, the magnitude is floored at 10^{-6} before exponentiation; at zero the result is zero. Protected division replaces a denominator whose magnitude is below 10^{-9} by a same-sign value of magnitude 10^{-9}, using the positive value at zero. The executable operators are:

Table 8: Core expression operators. Exponents r and a are fixed numerical constants; k, thresholds, and other arguments can be expressions containing fitted parameters.

Syntax	Meaning
`x+y`, `x-y`, `x*y`, `-x`	Addition, subtraction, multiplication, and negation
`x/y`	Protected division as defined above
`pow(x,r)`	$\mathrm{spow}(x, r)$
`sat(x,k,a)`	$\lvert x\rvert^a / \left((\lvert k\rvert + 10^{-9})^a + \lvert x\rvert^a + 10^{-12}\right)$
`exp(x)`	$\exp(\mathrm{clip}(x, -50, 50))$
`log1p(x)`	$\log(1 + \lvert x\rvert)$
`tanh(x)`, `sin(x)`	Hyperbolic tangent and sine
`step(x,th,f,g)`	f when $x > \mathrm{th}$, otherwise g

Signed power is distinct from ordinary even integer power: `pow(-2,2)` evaluates to -4, while `(-2)*(-2)` evaluates to 4. In particular, a Gaussian factor uses `exp(-x*x)`. The expression `pow(x,0)` is the sign function, with value zero at $x = 0$.

The parser also accepts the following operators and spellings:

Table 9: Additional accepted operators and spellings. Power aliases retain signed-power semantics; absolute value returns the ordinary magnitude.

Spelling	Meaning or expansion
`x**r` or `x^r`	`pow(x,r)`
`sqrt(x)`	`pow(x,1/2)`
`log(x)`	`log1p(x)`
`cos(x)`	$\sin(x + \pi/2)$
`sat(x,k)`	`sat(x,k,1)`
`abs(x)`	$\lvert x \rvert$

Absolute value can equivalently be expressed by `step(x,0,x,-x)`. The public grammar supplied to agents lists the core operators in Table 8.

D.4 Illustrative programs for the five evaluated classes

ODE with an unobserved state. The second state influences the observed dynamics without a direct measurement. Its initial value becomes the fitted parameter `z_0`; the observed channel provides the controllable coordinate.

```
class ODE
state x, z
latent z
input u
param a, b, c, k
d x/dt = -a*x + b*sat(z,k,2)
d z/dt = u - c*z - x*z
obs y1 = x
```

Discrete map. The right-hand side is the complete next state, with an externally selected input held fixed over a rollout.

```
class Map
state x
input u
param a, b, k
next x = a*x + b*tanh(u/k)
obs y1 = x
```

Stochastic differential equation. The two equations separately specify the drift and the state-dependent diffusion amplitude.

```
class SDE
state x
input u
param a, b, sigma, q, k
d x/dt = -a*x + b*tanh(u)
diffusion x = sigma + q*tanh(x/k)
obs y1 = x
```

Structural causal model. The input root `x` is connected to channel `y1`. The unobserved intermediate `z` is computed before `v`, whose readout is `y2`. An intervention on `y2` overrides the structural equation for `v`.

```
class StructuralDAG
state x, z, v
latent z
param a, b, c
let z = a*x
```

```
let v = b*tanh(z) + c*x*x
obs y1 = x
obs y2 = v
```

Static response function. The input readouts bind the queried field and temperature. The response can then vary with those settings through the declared expression.

```
class StaticFunction
state field, temperature, response
param a, b, c
let response = a*tanh(field/b) + c*temperature*temperature
obs field = field
obs temperature = temperature
obs signal = response
```

E DELAY, SPATIAL, STOCHASTIC-PROCESS, RESERVOIR, LARGER-NETWORK, AND TRANSFER DOMAINS

This appendix specifies the delay, spatial, stochastic-process, reservoir, 32-node network, and transfer domains of Table 2. Each uses the common experiment budget, accumulated observations, executable submissions, and held-out behavioral tests. The language adds five model classes: delay differential equations (`DDE`), spatial field equations (`PDEGrid`), continuous-time jump processes (`Jump`), microscopic update rules (`AgentRule`), and local-energy lattice dynamics (`LatticeMC`). Reservoir, larger-network, and transfer systems use the existing ODE class. Their held-out long-horizon tests last four times the standard run in the delay and spatial domains and three times in the jump, agent, and lattice domains, against ten times in the ODE and map domains.

E.1 DELAYED FEEDBACK

A delayed system depends on past states as well as its present state. The generator composes instantaneous and delayed responses, choosing the source coordinate, feedback sign, nonlinear response, and lag. A hypothesis can therefore distinguish a delayed influence from an additional instantaneous coupling. The expression `delay(x,tau)` denotes $x(t-\tau)$, where τ is constant during an experiment and lies in $[0.01, 8]$. Its value can be a fitted parameter.

Each experiment resets the full prehistory. The default history is constant at the preparation; a disclosed `history_slopes` context allows linear histories of controlled channels. Subsequent impulses and clamps are retained in delayed reads. The solver uses the method of steps, splitting intervals at interventions and their delayed effects. Clamps apply on half-open intervals. Each complete simulation permits 10^5 right-hand-side evaluations across its integration intervals; exhaustion is a numerical failure for that parameter trial. Before a world is accepted, a numerical check refines the integration and requires the delayed dependence to affect observable predictions. The following example proposes delayed feedback through an unobserved state:

```
class DDE
state x, z
latent z
input u
param a, b, c, tau
d x/dt = -a*x + b*tanh(delay(z,tau)) + u
d z/dt = c*(x-z)
obs y1 = x
```

E.2 SPATIALLY DISTRIBUTED FIELDS

The three PDE settings describe one-dimensional fields on a physical interval. Transport systems combine generated fluxes, diffusion, losses, forcing, and optional hidden storage. Reaction–diffusion systems combine nonlinear reactions, inter-field dependencies, and distinct diffusivities. Wave

systems evolve displacement and rate fields with composed restoring and damping responses, spatial stiffness that may depend on displacement, and optional memory. The generator chooses these responses as expression trees; the agent must construct the field equations.

The operators `dx(f)`, `dxx(f)`, and `diffuse(D,f)` denote $\partial_x f$, $\partial_{xx} f$, and $\partial_x(D\partial_x f)$, respectively. Expressions may depend on the fields, the physical coordinate `space`, and interval `length`; total spatial derivative order is at most two. A latent field begins uniformly at its fitted initial-value parameter, so increasing numerical resolution introduces no additional fitted coefficients. For example:

```
class PDEGrid
state c, r
latent r
input u
param D, v, a, k
d c/dt = -v*dx(c) + diffuse(D,c) - a*c*r + u
d r/dt = k*(c-r)
obs y1 = c
```

An acquisition's `context` specifies interval length, boundary conditions, initial profiles, forcing profiles, and sensor coordinates. Ordinary lengths range from 0.8 to 1.2; probes extend this to $[0.7, 1.7]$. Available boundaries are periodic, zero-gradient Neumann, and zero-value Dirichlet. Preparations and forcing use constant, sinusoidal, or Gaussian profiles. Sinusoids have modes one through four and a chosen phase; Gaussian centers and widths are specified as fractions of interval length, with widths in $[0.08, 0.3]$. Periodic Gaussians are smoothly periodized. An optional profile offset lies in $[-1, 1]$. A pulse profile specifies the shape of an impulse or whole-field clamp.

The agent chooses four to 32 physical sensor locations. Responses return aligned time, position, and channel rows, retaining early-time measurements as well as later checkpoints. Acquisition price scales with sensor count and duration. Sensor density is distinct from the evaluator's numerical grid: an agent purchases observations, not integration cells.

The simulator uses second-order cell-centered spatial approximations, linear sensor reconstruction, and sparse adaptive backward-differentiation integration with relative and absolute tolerances 10^{-7} and 10^{-9}. Ordinary simulations use 64 cells and hidden tests use 96–256. Before a world is accepted, a check doubles the numerical resolution and compares predictions at the same physical sensors, requiring a discrepancy below 0.01 NRMSE for every held-out condition. Removing spatial operators must materially alter observable behavior. Hidden experiments vary preparations, boundaries, physical length, controls, duration, and numerical resolution.

A simulation permits at most 10^5 right-hand-side evaluations, including numerical Jacobian probes and all intervention segments. Exceeding this allowance marks the parameter trial as a numerical failure, allowing fitting to continue with other trials. Only requested observation checkpoints are retained. This limit on numerical work is separate from the agent's experiment budget.

E.3 Finite-activity jump processes

Jump systems represent discrete events with state-dependent rates. The generator varies nonlinear rate expressions, inhibition, branching, conversion, and joint events. The agent supplies each event's rate and simultaneous state increments. For example, reproduction, loss, and conversion are expressed as:

```
class Jump
state n, m
param birth, death, transfer
rate reproduce = birth*n
change reproduce n = 1
rate loss = death*n
change loss n = -1
rate convert = transfer*n
change convert n = -1
change convert m = 1
obs y1 = n
obs y2 = m
```

Rates depend on current states and constant inputs. Event increments are evaluated on the pre-event state and applied together. The Gillespie direct method samples event times exactly for this finite-activity class; negative or nonfinite rates invalidate a simulation. Preparations and clamps set nonnegative integer counts, and impulses introduce positive integer counts. Ordinary durations lie in $[0.5, 8]$, with probes extending to twelve. The task is to recover event mechanisms from ensembles of count trajectories, rather than match individual random realizations.

E.4 MICROSCOPIC AGENT RULES

The microscopic setting asks for local update rules that explain population-level measurements. Individuals occupy nodes of a selected directed graph. The generator varies nonlinear response composition, aggregation order, persistent individual types, and hidden memory. A program specifies synchronous updates and optional independent Gaussian noise amplitudes. The operator `neighbors(f)` averages f over incoming neighbors, returning zero for an isolated node; `mean(f)` and `variance(f)` summarize the population. For example:

```
class AgentRule
state opinion, memory
latent memory
input u
param a, b, c, r, sigma
types 1
next opinion = tanh(a*opinion+b*neighbors(opinion)-c*memory+u)
next memory = r*memory+(1-r)*opinion
noise opinion = sigma
obs y1 = mean(opinion)
obs y2 = variance(opinion)
```

The declaration `types K` allows one to five persistent types. Each individual's unobserved `kind` is drawn uniformly from $0, \ldots, K-1$ and can enter its update rule. Noise draws are independent across individuals, states, and updates. Initial individual offsets are centered on the selected population mean; hidden states share a fitted initial value.

Experimental context selects graph edges, population size, initial spread, and intervention nodes. Ordinary populations contain eight to 48 individuals and probes reach 96. Runs last two to 32 updates, or up to 48 in probes. In addition to aggregate observations, a `deep_assay` purchases one individual's measured-state trace from an acquired path, priced per recorded time. The trace supplies evidence about microscopic behavior without revealing the individual's hidden type. As with scalar internal assays, it informs the agent's reasoning while the standard fitter uses the aggregate observations.

E.5 LOCAL-ENERGY LATTICE SYSTEMS

The lattice setting asks an agent to infer local interactions from collective stochastic dynamics. A program specifies a local energy expression on a periodic square lattice of spins $s \in \{-1, +1\}$. The total Hamiltonian sums this expression over all lattice anchors. The operator `shift(f,i,j)` translates a local expression by integer offsets $i, j \in [-4, 4]$. The generator samples interaction supports, orders from two to five spins, signs, strengths, and external-field responses, withholding the interaction list. An illustrative hypothesis is:

```
class LatticeMC
state s
input temperature, field
param J, K
energy = -J*s*shift(s,1,0)-K*s*shift(s,0,1)*shift(s,1,1)-field*s
obs y1 = mean(s)
obs y2 = mean(s*shift(s,1,0))
```

Each sweep makes L^2 random-site visits with replacement on a side-L lattice. A visit proposes a spin flip with probability one half and accepts it according to the Metropolis rule using the full Hamiltonian change and bath temperature. The lazy proposal removes parity periodicity. Initial

magnetization sets independent spin probabilities; clamps pin selected sites to either spin value. Measurements include transients rather than presupposing equilibrium. Ordinary side lengths range from four to ten and probes reach sixteen; runs last one to twelve sweeps, extending to 24 in probes. Unlike PDE grid density, lattice size is a physical system-size choice.

E.6 Stochastic acquisition and evaluation

Jump, agent, and lattice acquisitions purchase one to 256 independent paths, subject to any disclosed per-action allocation limit. Measurements of the simulated counts and aggregates are exact; randomness arises from the process itself. Acquisition, fitting, and hidden evaluation use separate seeded random streams. Prices scale with path count and duration, additionally with population size or lattice area for microscopic systems.

The fitter minimizes ensemble discrepancies over channel distributions, pairwise channel sums, and increments, using fixed fitting streams. It uses the derivative-free Nelder–Mead method with six random starts, or one supplied starting point, and sixty objective evaluations per start. Hidden evaluation uses 512 independent reference and candidate paths per condition. Error is normalized energy/CRPS excess over the disclosed projections and six time checkpoints; each test group must remain below 0.04. Native distributional errors are retained alongside the binary outcome. These comparisons assess predictive distributions, not pathwise agreement between independent random samples.

E.7 Reservoirs and larger networks

The reservoir setting samples directed exchanges between two observed stores and two hidden stores, including a slow store. Internal transfer terms conserve total stored material, while recharge and spill change it. Both the exchange graph and the composed flux functions are unknown. Agents use the same preparations, interventions, scalar internal assays, and numerical fitter as in the other ODE settings.

Network-S2 extends the sparse network construction to 32 measured nodes. A randomly ordered directed cycle provides the backbone, with four to nine additional edges. Five shared fitted coefficients govern the node and edge responses. Its disclosed expression limit is 1,200 nodes, accommodating the larger graph; the sixteen-parameter limit is unchanged. The five- and twelve-node domains retain their separate generation distributions and limits.

E.8 Transfer problems

Transfer tests whether a discovered scientific mechanism helps explain a different system. Each study contains a first world, a related world, and a control world. The first and related worlds contain the same generated expression subtree, attached to different variables and embedded in different surrounding equations; their fitted coefficients are independent. The control world changes that subtree while matching the related world's remaining equations, coefficients, preparations, measurement-noise distribution, and hidden conditions. A study is accepted only if the subtree is literally shared and both removing and replacing it change behavior materially.

One agent first investigates the first world. Its last accepted program, canonically printed without comments, is given to the later instances that receive it; its fitted coefficient vector, observations, and conversation are not supplied. Four fresh investigations then compare agents given the program with agents starting without it, on the related and on the control world, in randomized order. The related and control worlds both receive the larger of their independently calibrated budgets, and each comparison with and without the program uses the same world, noise seed, and inference configuration. This paired design distinguishes reuse of a shared mechanism from a general benefit of seeing another submitted model.

E.9 Transfer results

Table 10 gives the paired transfer results summarized in Section 4.5.

Table 10: Paired transfer results. In each study, an agent given the first world's final program and an agent starting without it investigate the related world, which shares the mechanism, and the control world, which does not, with the same noise seed and budget. "Studies" counts the studies whose first investigation produced a submitted program (one GLM study did not), and "First world" those in which it was solved. Entries are solve counts.

			Related world		Control world	
Model	Studies	First world	Fresh	Given program	Fresh	Given program
GPT 6 Astra	4	1	2	2	1	0
Claude Fable 5.1	4	2	0	2	0	0
DeepSeek v4.1 Flash	4	0	0	0	0	0
Qwen 3.8 Max	4	1	1	1	0	0
Muse Spark 1.3	4	1	0	1	0	0
Kimi K3	4	0	1	0	0	0
Grok 4.6	4	0	0	0	0	0
GLM 5.3	3	1	0	0	0	0
Gemini 3.8 Flash	4	0	0	0	0	0

F MODEL CONFIGURATIONS

All runs use the same experimental interface and fitting harness. The highest supported reasoning setting is `xhigh` for Qwen, `high` for Gemini, and `max` for the other models. Sampling temperature, top-p, and provider seed use endpoint defaults. The context and output allowances are listed below; conversation history is retained until the agent finishes or an endpoint limit ends the episode.

Table 11: Endpoint configurations. Token counts are context/output ceilings.

Model	Provider	Context	Output	Effort	Cache
GPT 6 Astra	OpenAI	1,050,000	128,000	`max`	—
Claude Fable 5.1	Anthropic	1,000,000	128,000	`max`	1h
GLM 5.3	Z.AI	1,048,576	131,072	`max`	—
Grok 4.6	xAI	500,000	450,000	`max`	—
Muse Spark 1.3	Meta	1,048,576	943,718	`max`	—
Qwen 3.8 Max	Alibaba	1,000,000	131,072	`xhigh`	—
Gemini 3.8 Flash	Google AI Studio	1,048,576	65,536	`high`	—
Kimi K3	Moonshot AI	1,048,576	943,718	`max`	—
DeepSeek v4.1 Flash	Fireworks	1,048,576	943,718	`max`	—

Table 12: Model identifiers used for inference.

Label	OpenRouter identifier
GPT 6 Astra	`openai/gpt-6-astra`
Claude Fable 5.1	`anthropic/claude-fable-5.1`
GLM 5.3	`z-ai/glm-5.3`
Grok 4.6	`x-ai/grok-4.6`
Qwen 3.8 Max	`qwen/qwen3.8-max-0902`
Muse Spark 1.3	`meta/muse-spark-1.3`
Gemini 3.8 Flash	`google/gemini-3.8-flash`
Kimi K3	`moonshotai/kimi-k3`
DeepSeek v4.1 Flash	`deepseek/deepseek-v4.1-flash`

G CONTINUOUS ERROR DEFINITIONS

For each deterministic test group τ, concatenate the generating system's noiseless trajectories and define the channel normalizer

$$n_{\tau j} = \mathrm{std}(y^{\star}_{\tau j}) + 0.05\,\mathrm{mean}\,|y^{\star}_{\tau j}| + 10^{-9}. \tag{8}$$

For each hidden condition, compute timewise RMSE on each real channel, divide by $n_{\tau j}$, and average across channels. The group error E_τ averages these condition-level errors; Equation 3 takes the

worst group. Divergent simulations receive an internal failure penalty, which is separated from finite prediction errors in the reported continuous summaries. Missing submissions and invalid predictions remain in the evaluation denominator.

For stochastic systems, the evaluator simulates independent predicted and reference path ensembles with fixed evaluation seeds. Distributional error uses normalized energy/CRPS excess, and first-passage error uses excess Brier loss. The maximum excess over the corresponding hidden conditions determines each reported error. A stochastic world passes when both are below their respective thresholds, 0.04 and 0.03.

H PER-WORLD ERRORS AND CALIBRATION ERRORS

Table 13: Worst-group NRMSE for every deterministic model–world pair in the domains marked † in Table 2. World numbers identify the same paired instances across models. D, I, and N are the failure categories of Figure 4: fails to simulate, missing channel or invalid predictions, and no submission.

Setting	World	GPT	Claude	GLM	Grok	Muse	Qwen	Gemini	Kimi	DeepSeek
Mechanics	1	0.458	0.389	0.259	0.597	0.357	0.469	0.495	0.844	0.369
	2	0.005	0.005	0.056	0.073	0.005	0.005	0.210	0.230	I
	3	0.293	0.208	0.130	0.512	0.432	0.210	0.542	0.633	0.730
	4	3.063	3.037	3.704	D	1.106	3.806	3.942	3.082	3.948
Kinetics	1	D	0.429	D	0.659	0.854	0.400	0.956	0.515	0.355
	2	0.300	D	1.405	D	109.923	0.273	2.491	0.365	D
	3	0.691	0.813	0.866	0.849	D	0.736	0.748	0.854	0.660
	4	0.030	0.208	0.476	12.982	1.659	0.506	D	1.116	2.494
Discrete	1	0.061	0.017	0.050	0.369	0.281	0.074	D	0.433	0.402
	2	0.364	0.276	0.491	0.495	0.631	0.502	0.527	D	D
	3	0.019	0.226	0.285	0.765	0.598	0.279	0.494	D	0.576
	4	1.749	1.109	3.214	1.825	0.531	4.580	D	1.475	35.185
Causal	1	0.002	0.005	0.007	0.018	0.043	0.261	0.148	0.092	0.256
	2	0.012	0.014	0.017	0.062	0.094	0.116	0.005	0.128	0.156
	3	0.014	0.014	0.025	0.018	0.024	0.127	0.036	0.130	0.287
	4	0.127	0.108	N	0.129	0.147	N	0.286	0.186	0.489
Scaling	1	0.258	0.530	0.168	2.333	0.975	0.354	0.296	0.447	3.773
	2	0.020	0.017	0.363	0.363	0.363	0.363	0.029	0.419	1.366
	3	0.409	0.882	0.336	0.183	0.098	0.334	0.434	0.536	1.080
	4	0.064	0.449	0.448	0.449	0.952	0.725	6.451	0.447	0.268
Ecology	1	0.091	0.087	0.094	0.171	0.195	0.066	0.722	0.391	0.230
	2	0.267	0.158	1.225	0.333	0.152	D	0.292	0.159	2.219
	3	0.153	1.675	0.171	1.228	0.806	0.131	1.104	0.602	D
	4	0.113	0.334	0.072	0.152	0.099	0.059	0.131	0.132	0.203
Network	1	0.278	0.129	0.519	0.186	0.540	0.579	0.719	0.450	0.266
	2	0.088	0.069	0.254	0.075	0.202	0.732	0.139	0.144	0.422
	3	0.061	0.076	0.270	0.118	0.283	0.732	0.263	0.321	0.243
	4	0.214	0.204	0.272	0.320	0.744	0.491	0.400	1.001	0.335
Network-S1	1	0.094	0.105	0.312	0.478	0.181	0.585	0.732	0.779	0.966
	2	0.023	0.281	0.314	0.573	0.647	N	0.663	0.238	1.782
	3	0.005	0.305	0.144	0.466	0.234	0.195	1.175	0.431	1.073
	4	0.021	0.051	0.610	0.651	0.201	5.351	12.890	10.634	0.707
Metrology	1	1.147	0.179	0.876	0.282	0.759	I	2.319	0.491	0.308
	2	0.103	0.091	0.184	0.157	2.162	0.158	0.480	I	I
	3	0.994	0.136	0.171	0.359	0.219	0.210	I	0.250	0.161
	4	1.288	I	I	0.491	0.129	0.090	I	I	0.330
Spectroscopy	1	0.012	0.009	I	0.076	0.025	0.012	0.056	0.013	I
	2	0.009	0.015	0.008	0.023	0.015	I	0.028	0.019	I
	3	0.012	0.088	0.072	0.231	0.080	0.145	0.249	I	0.111
	4	0.012	0.016	0.022	0.074	0.141	0.023	0.060	I	I

Table 14: Stochastic prediction errors in their native score units. Each world has a maximum energy/CRPS excess and maximum first-passage Brier excess; thresholds are 0.04 and 0.03. A dash denotes no submission.

Model	1: CRPS	1: Brier	2: CRPS	2: Brier	3: CRPS	3: Brier	4: CRPS	4: Brier
GPT 6 Astra	0.1122	0.0004	0.0327	0.0316	0.0090	0.0014	0.0144	0.0015
Claude Fable 5.1	0.0857	0.0006	0.0563	0.0011	0.0236	0.0037	0.0282	0.0006
GLM 5.3	—	—	—	—	—	—	0.0294	0.0017
Grok 4.6	—	—	0.3768	0.0009	0.0377	0.0030	0.0116	0.0011
Muse Spark 1.3	0.0710	0.0007	0.0832	0.0003	0.0087	0.0018	0.0426	0.0010
Qwen 3.8 Max	0.0548	0.0009	—	—	0.0088	0.0020	0.0338	0.0014
Gemini 3.8 Flash	0.4257	0.0007	0.5982	0.0011	0.0196	0.0011	0.0741	0.0017
Kimi K3	0.0848	0.0007	0.4539	0.0010	0.0650	0.0018	—	—
DeepSeek v4.1 Flash	0.0655	0.0005	0.0568	0.0012	0.0133	0.0026	0.0140	0.0020

Table 15: Calibration errors for deterministic worlds (Section 2.6). Each entry is the worst NRMSE across the five further noise seeds at the assigned budget, fitting the true program from its true coefficients on a design spanning the held-out regimes. The known-structure baseline of Section 4.1 instead starts from random coefficients on the random design. Values retain the precision recorded during calibration.

Setting	World 1	World 2	World 3	World 4
Mechanics	0.0087	0.0023	0.0471	0.0115
Kinetics	0.0738	0.0169	0.0751	0.0129
Discrete	0.0103	0.0575	0.0159	0.0069
Causal	0.0651	0.0372	0.0294	0.0565
Scaling	0.0474	0.0570	0.0144	0.0223
Ecology	0.0569	0.0196	0.0178	0.0125
Network	0.0061	0.0071	0.0113	0.0175
Network-S1	0.0041	0.0016	0.0080	0.0062
Metrology	0.0078	0.0100	0.0171	0.0073
Spectroscopy	0.0076	0.0090	0.0119	0.0200

I CLASSICAL COMPARISONS

Table 16 compares methods on their shared subsets of the 44 worlds in the domains marked † in Table 2. Each entry gives the pass count and median worst-group NRMSE; simulation failures and missing predictions enter the median as $+\infty$. The deterministic panel comprises 24 dynamical worlds, twelve static-function worlds, and four causal worlds. All classical methods on a given world share the same random-acquisition dataset, while language models choose their own experiments under the same acquisition budget.

Table 16: Classical and language-model results on matched world subsets. Dashes denote settings outside a method's scope. All 512 grammar proposals are evaluated in each of its forty worlds.

Method	Dynamical	NRMSE	Static	NRMSE	All deterministic	NRMSE
GPT 6 Astra	11/24	0.133	6/12	0.084	20/40	0.099
Claude Fable 5.1	6/24	0.217	6/12	0.113	15/40	0.168
GLM 5.3	4/24	0.299	3/12	0.260	10/40	0.278
Grok 4.6	2/24	0.504	3/12	0.256	8/40	0.361
Muse Spark 1.3	2/24	0.481	4/12	0.180	9/40	0.282
Qwen 3.8 Max	4/24	0.497	3/12	0.272	7/40	0.358
Gemini 3.8 Flash	0/24	0.720	4/12	0.365	6/40	0.511
Kimi K3	0/24	0.559	2/12	0.469	3/40	0.449
DeepSeek v4.1 Flash	0/24	0.718	0/12	1.223	0/40	0.683
Observed-state SINDy	0/24	∞	—	—	—	—
PySR	—	—	3/12	0.146	—	—
Grammar search	0/24	2.192	0/12	0.438	0/40	1.439
Known structure	16/24	0.015	8/12	0.020	27/40	0.019

PySR's median error on the static worlds is 0.146, compared with 0.084 for GPT and 0.113 for Claude. The known-structure baseline attains median NRMSE 0.019 over the 40 deterministic worlds. Its remaining failures reflect estimation from the random design with random starting coefficients, whereas calibration uses a design spanning the held-out regimes and starts from the true coefficients. The stochastic known-structure results are:

Table 17: Known-structure baseline on stochastic worlds, using the shared random design and random starting coefficients.

World	CRPS excess	Brier excess	Pass
1	0.0170	0.0005	Yes
2	0.0571	0.0012	No
3	0.0084	0.0020	Yes
4	0.0228	0.0011	Yes

Fitter ceiling. To test whether the agents' failures reflect the fitter's budget rather than their structures, we refit every executable but failing final deterministic model of these 44 worlds (248 episodes) on the agent's own purchased data, reproducing each archived training error exactly, with 24 starts of at most 200 residual evaluations (the protocol uses 6 starts of 60) and the submitted coefficients as one of the starts. None passes the held-out tests; 12 improve their training error by more than 10%, and 5 their held-out error. Fitting the true program on each world's calibration design from random starts passes 24 of the 40 deterministic worlds with the protocol fitter and 26 with 24 starts of 200 evaluations, against all 40 from the true coefficients and 27 on the random design (Table 16). Of the 16 protocol failures, 13 end above the truth's training error, so local minima rather than the design limit the known-structure baseline; the other three match it with coefficients the design does not pin down. The agents' submitted structures, by contrast, gain nothing from the larger budget.

J Paired error comparisons

J.1 Final-outcome categories

Table 18: Error profiles across the 44 worlds of the domains marked † in Table 2. Categories are disjoint. "Finite error" denotes predictions exceeding a scoring threshold. "Simulation" denotes at least one divergent, nonfinite, or shape-invalid hidden prediction. "Missing channel" denotes an omitted required readout. An episode without a submission is classified separately.

Model	Pass	Finite error	Simulation	Missing channel	No submission
GPT 6 Astra	22	21	1	0	0
Claude Fable 5.1	17	25	1	1	0
GLM 5.3	11	26	1	2	4
Grok 4.6	10	31	2	0	1
Muse Spark 1.3	10	33	1	0	0
Qwen 3.8 Max	9	29	1	2	3
Gemini 3.8 Flash	7	32	3	2	0
Kimi K3	3	34	2	4	1
DeepSeek v4.1 Flash	2	34	3	5	0

J.2 Spectroscopy: executable verification and response to feedback

The four models in Table 19 investigate the same world with a 138-unit budget and finish voluntarily. Each final program is replayed at the settings in its own training acquisitions, using the exact submitted parameter vector. This replay reproduces the recorded fitting cost. Kimi's constant prediction spans observations from 0.0311 to 0.1966; DeepSeek's zero prediction spans observations from 0.0305 to 0.1759.

Table 19: Paired spectroscopy comparison. Measurements are those available at the final submission. The prediction range is evaluated on those measurements; hidden error is worst-group NRMSE. The Kimi and DeepSeek programs omit the input/readout bindings and therefore ignore the experimental settings.

Model	Measurements	Fit cost	Predicted intensity range	Hidden error
GPT 6 Astra	157	0.00398	[0.0354, 0.2037]	0.0121
Claude Fable 5.1	162	0.00323	[0.0346, 0.1941]	0.0157
Kimi K3	192	2.13896	[0.0340, 0.0340]	Missing readouts
DeepSeek v4.1 Flash	222	4.81083	[0.0000, 0.0000]	Missing readouts

The binding rule appears in the supplied language description: a direct readout links a channel to a state, so an intervention on that channel acts on that state. For static inputs, the following declarations connect the measured settings to the program:

```
obs frequency = fq
obs field = fi
obs temperature = te
obs resolution = re
```

Claude first tests a minimal binding example, identifies the required declarations, and refits its spectral model on the same 79 observations. Its training error falls from 0.83490 for the initial fit without these declarations to 0.01368 after adding them. Kimi instead concludes that the harness evaluates a "fixed zero context," pins its coefficients, and finishes with the claim "The model is confirmed." DeepSeek similarly attributes the uninformative fits to the fitting procedure and substitutes a hand-derived expression. The distinguishing behavior is whether contradictory diagnostics lead to a corrected executable.

DeepSeek's expression also illustrates a mismatch between mathematical intent and execution. Its prose describes a Gaussian factor, but the submitted expression uses `exp(-pow(z,2))`. The disclosed signed-power convention makes this factor $e^4 \approx 54.60$ at $z = -2$, whereas a Gaussian square gives $e^{-4} \approx 0.0183$. Explicit multiplication, as used by GPT and Claude, preserves the intended square. This semantic discrepancy is identifiable directly from the submitted expression and the public operator definition.

J.3 Network: identical recovered topology, different governing equations

Mapping states to observed channels verifies that GPT's and DeepSeek's final programs both contain exactly the generating graph's sixteen directed edges. Their fourth-node equations illustrate the remaining difference:

$$\text{DeepSeek:} \quad \dot{x}_4 = -ax_4 + wu(x_5 + x_8 - 2x_4) + c_4, \tag{9}$$

$$\text{GPT:} \quad \dot{x}_4 = ax_4 - bx_4^3 + gu\big[\tanh(x_5 - x_4) + k\tanh(x_8 - x_4)\big] + h. \tag{10}$$

DeepSeek tests one saturating-coupling variant but retains an affine final model. GPT combines saturation with nonlinear self-dynamics, including different response forms across nodes. Both identify the relevant neighbors; their scientific hypotheses differ in the responses assigned to those interactions. The final errors, 0.9655 and 0.0942, make that distinction consequential under held-out conditions.

K Submitted structures in the case studies

The following programs are the actual submitted structures. Their parameters are estimated by the shared fitting harness; the main text reports the relevant fitted coefficients and observed errors.

K.1 DeepSeek: ecology

The first submitted program is:

```
class ODE
state a, b, c1, c2
latent c1, c2
input u
param p1, q1, r1, s1, p2, q2, r2, s2, kk, g2
d a/dt = -(p1 + q1/u)*a - r1*a*a - s1*a*a*a
d b/dt = -(p2 + q2/u)*b - r2*b*b - s2*b*b*b
d c1/dt = kk*(b - c1) + g2*(b - c1)*step(t, 0.67, 0, 1)
d c2/dt = kk*(c1 - c2) + g2*(c1 - c2)*step(t, 0.67, 0, 1)
obs y1 = a
obs y2 = b
obs y3 = c2
```

The fourth and final program is:

```
class ODE
state a, b, c1, c2
latent c1, c2
input u
param p1, q1, v1, r1, p2, q2, v2, r2, kk, g2, w1, w2, w4
d a/dt = -(p1 + q1/u + v1*(a/u))*a - r1*a*a
d b/dt = -(p2 + q2/u + v2*(b/u))*b - r2*b*b
d c1/dt = kk*(b - c1) + g2*(b - c1)*step(t, 0.67, 0, 1)
d c2/dt = kk*(c1 - c2) + g2*(c1 - c2)*step(t, 0.67, 0, 1)
obs y1 = a
obs y2 = b
obs y3 = c2 + (w1 + w2*step(t, 0.67, 0, 1))*(b - c2) + w4*(b - c1)
```

The earlier second-population equation contains a negative cubic term. In the final model, its quadratic coefficient is $0.189415/u - 0.190566$, yielding the unstable low-input dynamics analyzed in the main text. The final observation map also introduces a gated blend of measured and latent states.

K.2 GROK: MECHANICS

The second submission is:

```
class ODE
state x, u, x2
latent x, u
param w2, g, k, k3, p, q, c
d x/dt = u
d u/dt = -w2 * x - g * u + p * x2 * x2
d x2/dt = -k * x2 - k3 * x2 * x2 * x2 + q * x
obs y1 = x
obs y4 = x
obs y2 = x2
obs y3 = c
```

The third and final submission is:

```
class ODE
state x, u, x2
latent x, u
param w2, g, k, k3, p, ks, q, c
d x/dt = u
d u/dt = -w2 * x - g * u + p * sat(x2 * x2, ks, 1)
d x2/dt = -k * x2 - k3 * x2 * x2 * x2 + q * x
obs y1 = x
obs y4 = x
obs y2 = x2
obs y3 = c
```

The respective OOD, intervention, and long-horizon errors are $(0.3935, 0.2276, 0.1780)$ and $(0.5118, 0.2555, 0.1638)$. The revised saturation improves long-horizon prediction but worsens both other groups.

K.3 MUSE: STOCHASTIC DYNAMICS

Both submissions use the same structure:

```
class SDE
state x1, x2
input u
param a, p, q, k1, s0, s1, c0, c1, k, au1, bu2
d x1/dt = -a*x1 + au1*u
d x2/dt = p*pow(x1, 0)*sat(x1, k1, 2) + q*x2 + bu2*pow(u, 0)*u
diffusion x1 = s0 + s1*x1
diffusion x2 = c0 + c1*pow(x2, 0)*sat(x2, k, 1)
obs y1 = x1
obs y2 = x2
```

The language defines signed power as $\mathrm{pow}(x, r) = \mathrm{sign}(x)|x|^r$; hence $\mathrm{pow}(x, 0)$ represents the sign function. Muse submits this program after 31 experiments and again, refitted, after 44. The first submission solves the world with a maximum excess CRPS of 0.023; the refitted final one fails at 0.071, above the 0.04 threshold, with the structure unchanged.

K.4 GPT: TWELVE-NODE NETWORK

The acquisition sequence contains twelve single-node clamps, four long-regime probes, two input changes, three initial-condition experiments, and one ordinary observation. The final program is:

```
class ODE
state x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11, x12
input u
param a, b, c, d, g, h, k, a1, b1, q
d x1/dt = a1*x1-b1*pow(x1,2)+g*u*(k*tanh(x7-x1)+q*tanh(x9-x1))-h
d x2/dt = a*x2-b*pow(x2,3)+g*u*tanh(x1-x2)+h
d x3/dt = a*x3-b*pow(x3,3)+g*u*tanh(x2-x3)+h
d x4/dt = a*x4-b*pow(x4,3)+g*u*(tanh(x5-x4)+k*tanh(x8-x4))+h
d x5/dt = a*x5-b*pow(x5,3)+g*u*tanh(x12-x5)-h
d x6/dt = a*x6-b*pow(x6,3)+g*u*(k*tanh(x2-x6)+tanh(x11-x6))+h
d x7/dt = a*x7-c*pow(x7,2)+g*u*tanh(x10-x7)-h
d x8/dt = a*x8-b*pow(x8,3)+g*u*(tanh(x6-x8)+k*tanh(x10-x8))+h
d x9/dt = a*x9-b*pow(x9,3)+g*u*tanh(x4-x9)+h
d x10/dt = a*x10-c*pow(x10,2)+g*u*tanh(x3-x10)+h
d x11/dt = a*x11-d*pow(x11,1.5)+g*u*tanh(x7-x11)+h
d x12/dt = a*x12-b*pow(x12,3)+g*u*tanh(x8-x12)+h
obs y1 = x1
obs y2 = x2
obs y3 = x3
obs y4 = x4
obs y5 = x5
obs y6 = x6
obs y7 = x7
obs y8 = x8
obs y9 = x9
obs y10 = x10
obs y11 = x11
obs y12 = x12
```

L ECOLOGY PROMPT AND OBSERVATION EXAMPLE

The system message introduces the task as follows:

> You are the participant in the benchmark described by the JSON below. Use the supplied JSON tools to interact with it. Large action responses may name an archive entry; response_archive retrieves the exact response.

It then supplies the domain narrative, channels, budget, controllable settings, acquisition actions, hypothesis grammar, scoring instructions, and resource policy. Selected fields from the ecology example are:

```
{
  "domain": "ecology",
  "channels": ["y2", "y1"],
  "budget": 243.0,
  "inputs": {
    "u": {"box": [0.3, 1.7], "default": 1.5858456350818932}
```

```
  },
  "submission": {
    "classes": ["ODE", "Map", "StructuralDAG", "StaticFunction", "SDE"],
    "interface": "program text (see grammar) or Python model class",
    "max_ast_nodes": 400,
    "max_params": 16
  }
}
```

Neither the illustration nor the hidden structure of Figure 1 is part of the prompt; the number of resources and their equations are withheld. The interface line also names a Python model class, which the JSON tools cannot carry; every submission in the evaluation was program text. The ranking rule quoted to the agent orders an episode's own diagnostics; the paper reports only the pass outcome and the worst-group error. The user message instructs the agent to begin, use the available actions, and call `finish` when finished.

The first archived action is

```
act({"action":{"type":"observe"}})
```

Its response contains 25 observation rows at times from zero to eight, with channels ordered `["y2","y1"]`, and the updated remaining budget. Subsequent hypotheses are supplied through `fit` and `submit`; complete measurements remain accessible through the observation archive.

The seven experiment types of this world, with their prices in budget units and their argument ranges, are: `observe` (2), a standard run; `set_initial` (3), starting populations in $[0.15, 1.25]$; `kick` (6), an impulse of size 0.165–0.825 at a time in $[0, 8]$; `clamp` (8), a held value in $[0.05, 2.5]$ over a window in $[0, 8]$; `set_input` (10), a feed rate $u \in [0.3, 1.7]$; `probe_regime` (14), starting populations in $[0.05, 2.5]$ and optionally three times the standard duration; and `deep_assay` (15), one latent state at one time of the most recent run. The standard preparation is $\mathtt{y1} = 0.26$, $\mathtt{y2} = 0.67$ with $u = 1.59$, and a standard run records 25 times over a horizon of 8.

GPT's episode on this world (Section 4.4) consists of 18 experiments (one standard run, five chosen preparations, two input changes, two impulses, four clamps, and four probes), seven assays, 32 fits, three logged hypothesis sets, and one submission. Its final two calls, verbatim except for rounding and elisions (. . .), are:

```
LLM  act({"action": {"type": "submit", "init": [1.24, 0.36, ...], "bounds": [[0, 5], ...],
      "program": "class ODE
      state x, z, r, s
      latent r
      latent s
      input u
      param b, c, q, p, e, f, A, h, v, B, k
      d x/dt = c*sat(x,q,1)+x*(r-b)
      d z/dt = z*(s-e-f*pow(z,2))
      d r/dt = A*u-(h+v*x+p*z)*r
      d s/dt = B*u-(k+v*x+p*z)*s
      obs y1 = x
      obs y2 = z"}})
tool {"ok": true, "accepted": true, "cost": 0.0, "remaining": 1.0, "experiments_run": 18,
      "params": ["b", "c", "q", "p", "e", "f", "A", "h", "v", "B", "k", "r_0", "s_0"],
      "fit": {"theta": [1.239, 0.356, 0.947, 0.009, 0.372, 0.141, 0.080, 0.130, 0.012, 0.028,
                        0.199, 0.620, 0.196],
              "cost": 0.00063, "per_sample_rmse": [0.028, 0.027, 0.026, ..., 0.024],
              "n_starts": 6, "success": true}}
LLM  finish({})
```

The benchmark fits the eleven declared coefficients and the initial values `r_0` and `s_0` of the two latent resources on the 18 purchased runs and returns them with the training error and one RMSE per run. The hidden test error of this model, $E = 0.091$, is computed after the episode and is never shown to the agent.

M PER-DOMAIN RESULTS, TERMINATION, AND COST

Table 20: Worlds solved out of four per domain, by agent.

Domain	GPT	Claude	Muse	Grok	Qwen	GLM	DeepSeek	Gemini	Kimi	All
Dynamics										
Mechanics	1	1	1	1	1	1	0	0	0	6
Kinetics	1	0	0	0	0	0	0	0	0	1
Discrete	2	1	0	0	1	1	0	0	0	5
Ecology	1	1	1	0	2	2	0	0	0	7
Network	2	2	0	1	0	0	0	0	0	5
Network-S1	4	1	0	0	0	0	0	0	0	5
Network-S2	2	0	0	0	0	0	0	0	0	2
Reservoir	1	0	0	0	0	0	0	0	0	1
Delay	4	4	2	3	4	3	4	3	4	31
Spatial fields										
Transport	4	4	4	3	3	4	2	3	4	31
Reaction–diffusion	4	4	4	4	3	3	4	4	4	34
Wave	4	4	4	2	4	1	3	0	1	23
Static and causal										
Causal	3	3	3	3	0	3	0	2	1	18
Scaling	2	1	1	0	0	0	0	1	0	5
Metrology	0	1	0	0	1	0	0	0	0	2
Spectroscopy	4	4	3	3	2	3	0	3	2	24
Stochastic										
Stochastic dynamics	2	2	1	2	2	1	2	1	0	13
Jump	3	3	0	1	0	0	2	0	1	10
Agents	3	2	0	0	0	0	1	0	0	6
Lattice	1	0	0	0	0	0	0	0	0	1

Of the 36 episodes per domain, the most often solved are reaction–diffusion in 34, delay in 31, and transport in 31. The least often solved are kinetics, reservoir, and lattice, in one episode each, and the 32-node network, in two.

Table 21: Budget use, termination, model selection, and API cost over the 80 worlds. "Budget used" is the mean share of the experiment budget spent. "Truncated" counts episodes that ended at the model's context or output limit. "Final worse" counts episodes whose final model has a higher held-out error than an earlier submission, and "Solve lost" those in which an earlier submission solved the world and the final one does not; earlier submissions are scored only for this analysis. Cost, in USD, is the API charge reported by OpenRouter. Requests served by an endpoint that reports only token counts are priced from those counts, at the provider's published rates for Claude and GPT and at the rates OpenRouter charged for Kimi's other requests. These estimates make up about half of the three models' totals.

Model	Budget used (%)	Truncated	No submission	Final worse	Solve lost	Cost ($)
GPT 6 Astra	93	0	0	10	1	959
Claude Fable 5.1	84	1	0	34	2	1687
Muse Spark 1.3	88	7	4	27	1	618
Grok 4.6	77	3	2	23	0	397
Qwen 3.8 Max	66	40	7	34	0	498
GLM 5.3	83	20	11	25	1	446
DeepSeek v4.1 Flash	90	3	2	32	0	43
Gemini 3.8 Flash	53	0	1	11	1	83
Kimi K3	90	12	1	39	1	729

Of the 720 episodes, 86 end at a context or output limit, and 14 of those still solve the world. Truncation also separates the agents: Qwen ends 40 of its 80 episodes at its context or output limit and GLM 20, whereas GPT and Gemini end none this way. On the SDE worlds, 29 of 30 submitted models predict the barrier-crossing probability within threshold but only 14 the distribution of the measured values: agents capture whether an event occurs far more often than the process's variability. Among the 321 executable deterministic models that fail, the worst-scoring test group is out-of-distribution in 154 cases and long-horizon in 145; interventions are worst in only 22.