SEARCH OR EXPERIMENT? BUDGET-AWARE METAREASONING FOR LLM-DRIVEN SCIENTIFIC DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are increasingly used to automate scientific discovery. Yet, existing approaches typically optimize hypothesis search over fixed evidence or active experimentation over newly acquired evidence in isolation. These operations have sharply asymmetric costs, since hypothesis refinement consumes computation, whereas experiments require substantial resources. Moreover, the marginal value of each action changes throughout discovery and must be inferred from the evolving discovery state. The central challenge is therefore to continually determine whether the next unit of budget is better spent refining what is already known or acquiring evidence that can change what is known. We introduce CoREAgent (Cost-optimized Refinement and Experimentation Agent), a budget-aware framework that casts scientific discovery as sequential allocation between SEARCH and EXPERIMENT. At each step, a learned metareasoner estimates the cost-adjusted downstream value of each action from the discovery state and chooses between refining hypotheses and acquiring evidence, while a separate acquisition policy decides where to experiment. CoREAgent learns these values online through prequential credit assignment, scoring each decision on evidence acquired after it was made. Across four interactive discovery benchmarks, CoREAgent improves exact accuracy by $8-19$ percent over the strongest baseline while reducing experimental usage by up to 31%. Our analyses isolate the contribution of each component and show that efficient discovery requires optimizing not only *what* to search for or experiment on, but *when* each action is worth its cost.[1]

1 INTRODUCTION

Large Language Models (LLMs) are increasingly deployed across scientific workflows, from hypothesis generation and model construction to experimental planning and autonomous laboratories (AI4Science & Quantum, 2023; Reddy & Shojaee, 2025; Zheng et al., 2025). Inspired by Wang & Buehler (2026)'s distinction between search within a fixed regime and evidence-driven revision of that regime, we model closed-loop discovery as a combination of *search*, which refines hypotheses using existing evidence, and *experimentation*, which acquires new evidence that can expand or revise the hypothesis space. Their costs are sharply asymmetric: search consumes computation and can be repeated cheaply, whereas each experiment consumes time, resources, and instrument access (Tom et al., 2024). This asymmetry creates a fundamental allocation problem where an agent must decide not only *which* experiment would be informative, but also *when* acquiring new evidence is more valuable than continuing to refine hypotheses under the current evidence. Choosing between search and experimentation requires a policy that jointly reasons over asymmetric costs, evolving relative returns, and the current evidence. A naive switching policy is inadequate since: (i) Search stagnation may reflect either an exhausted candidate pool or insufficient evidence to distinguish well-fitting mechanisms. (ii) When several hypotheses remain equally consistent with the observations, further optimization on the same data cannot break the tie; only new, discriminative evidence can (Box & Hill, 1967; Abhyankar et al., 2026b). (iii) Search is not free and consumes a finite computational budget through repeated LLM calls (Chen et al., 2024; Zhang et al., 2023). The value of each action is therefore state-dependent; search stagnates under insufficient evidence, while premature experimentation wastes scarce queries before informative distinctions emerge. The problem is therefore one of allocating search and experimental budgets according to their relative value throughout discovery, naturally motivating a metareasoning formulation (Russell & Wefald, 1991).

[1] https://anonymous.4open.science/r/core_agent

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

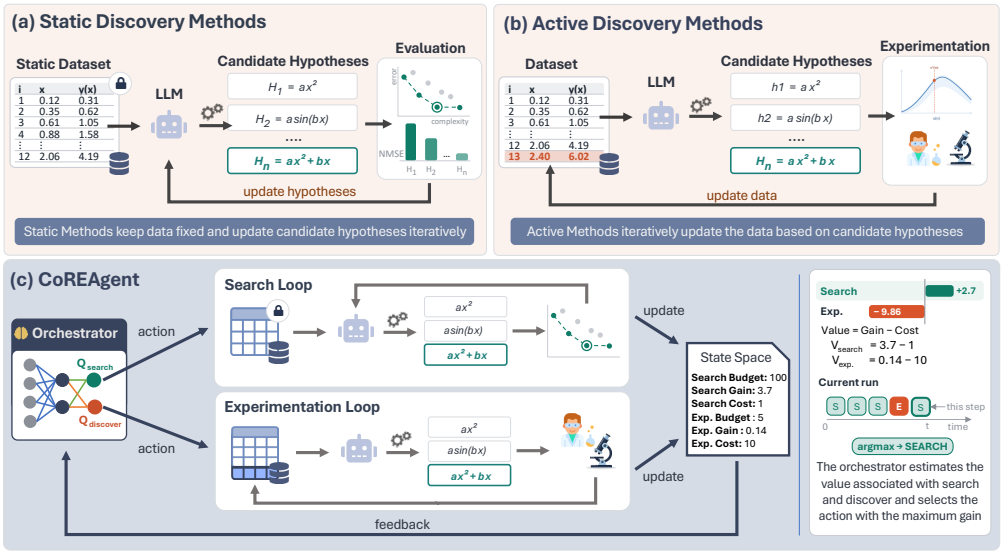


Figure 1: Overview of discovery paradigms. (a) **Static discovery methods** operate on a pre-collected, fixed dataset. (b) **Active discovery methods** acquire new observations iteratively, updating the dataset. (c) **CoREAgent** jointly orchestrates these two modes. A learned orchestrator selects between: *Search*: to improve hypotheses using the current data, and *Experiment*: to perform experiments to acquire informative new data. By estimating the state-dependent downstream value of consuming either resource, the orchestrator adaptively interleaves SEARCH and EXPERIMENT under separate computational and experimental budgets.

Existing methods leave this decision unresolved. As shown in Figure 1 (a), *static methods* such as LLM-SR, AlphaEvolve, and SR-Scientist strengthen hypothesis search using techniques like evolutionary search, agentic tool use, and reinforcement learning, but operate on fixed, pre-collected data (Shojaee et al., 2025; Novikov et al., 2025; Xia et al., 2026). On the other hand, *Active methods* (Figure 1 (b)) acquire evidence sequentially and optimize which experiment to run (Wiemann et al., 2026; Kabra et al., 2026; Murphy, 2026), but acquisition follows hypothesis updating on a fixed cadence since the experimental budget is optimized only *within* experimentation, never *between* search and experimentation. Classical decision theory weighs the expected value of information against acquisition cost (Lindley, 1956; Chaloner & Verdinelli, 1995; Rainforth et al., 2024), while metareasoning studies whether further computation is worth its cost (Russell & Wefald, 1991). LLM-driven discovery combines both decisions, but hypothesis search is generative and open-ended, making its future return difficult to characterize in advance. The unresolved question therefore is: *When is it worth spending scarce experimental budget to acquire new evidence rather than continuing hypothesis search over the evidence already available?*

To address this bottleneck, we introduce **C**ost-**o**ptimized **R**efinement and **E**xperimentation **Agent** (**CoREAgent**), *a budget-aware metareasoning framework that makes the search–experiment trade-off an explicit control problem*. At each step, the agent decides whether to exploit existing evidence through further hypothesis search or spend scarce experimental budget acquiring new evidence. SEARCH holds the evidence fixed and expands a structured population of candidate mechanisms through LLM-guided proposal, refinement, and cross-population exchange, while EXPERIMENT conducts the experiment expected to most reduce ambiguity among the current hypotheses and then updates the population with the new evidence. The metareasoner conditions on hypothesis quality and diversity, search progress, experimental opportunity, and the remaining search and experimental budgets, and learns the expected downstream value of consuming either resource at the current state. Because these values are not directly observable, CoREAgent estimates them retrospectively from how well earlier decisions improve predictive belief on subsequently acquired evidence. The resulting policy adapts the allocation of computation and experimentation throughout discovery rather than following a fixed search–acquisition schedule. We evaluate CoREAgent across different scientific discovery benchmarks, where it consistently improves mechanism recovery while consuming experimental resources more efficiently than other state-of-the-art baselines. Further analyses show that CoREAgent adapts its Search–Experiment allocation to changing action costs, while ablations isolate the contribution of feedback, diverse hypothesis search, adaptive metareasoning, and hypothesis-guided acquisition. Our contributions can be summarized as:

- **Budget-aware Scientific Discovery.** We formulate closed-loop *scientific discovery as a sequential control problem* over computation for refining hypotheses and costly experiments for acquiring new evidence. Under asymmetric costs and finite budgets, CoREAgent decides *when the expected value of a new experiment exceeds what further search can extract from the current evidence*.
- **Online metareasoning from future evidence.** We introduce a lightweight meta-controller that estimates the state-dependent, cost-adjusted value of SEARCH and EXPERIMENT, learning these values online through prequential credit from subsequently acquired evidence.
- **Cost-adaptive allocation.** The metareasoner systematically reallocates effort between SEARCH and EXPERIMENT as their relative costs change, demonstrating sensitivity to resource constraints.
- **Resource-efficient discovery.** Across four discovery benchmarks, CoREAgent improves accuracy by $8-19\%$ over the strongest baseline while reducing experimental usage by up to 31%.

2 BACKGROUND

2.1 POPULATION-BASED HYPOTHESIS SEARCH

Let $\mathcal{H}$ be a space of candidate mechanisms, each a model that predicts the outcome y of an experiment at condition $u \in \mathcal{U}$, and let $\mathcal{D}_t = \{(u_j, y_j)\}_{j=1}^{n_t}$ be the evidence so far. Hypothesis search seeks:

$$\hat{h} \in \arg\min_{h \in \mathcal{H}} \mathcal{L}(h; \mathcal{D}_t), \tag{1}$$

where $\mathcal{L}$ measures how well h explains $\mathcal{D}_t$. Population-based methods maintain candidates $\mathcal{P}_t = \{h_1, \ldots, h_K\}$ and evolve them iteratively by variation and selection (Rechenberg, 1973; Holland, 1975), keeping the evidence fixed as the population grows: $(\mathcal{D}_t, \mathcal{P}_t) \to (\mathcal{D}_t, \mathcal{P}_{t+1})$. When observations are sparse or non-informative, multiple structurally distinct mechanisms in $\mathcal{P}_t$ can fit $\mathcal{D}_t$ equally well and remain indistinguishable, necessitating new evidence to recover the governing law.

2.2 OPTIMAL EXPERIMENTAL DESIGN

The experimental design resolves this ambiguity among competing hypotheses by acquiring evidence under conditions where their predictions are most informative about the true mechanism. Since the true mechanism $h^\star$ is unknown, this uncertainty is represented by a *belief* $q_t(h) \approx p(h \mid \mathcal{D}_t)$ over the population $\mathcal{P}_t$. The Bayesian experimental design formalizes this by selecting the condition whose outcome Y_u is expected to be most informative about $h^\star$ (Box & Hill, 1967; Rainforth et al., 2024):

$$u_t^\star \in \arg\max_{u \in \mathcal{U}} \mathrm{I}\big(h^\star;\, Y_u \mid \mathcal{D}_t\big). \tag{2}$$

The observed outcome y_t results in the new evidence, $\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{(u_t^\star, y_t)\}$, while retaining the current hypothesis population $(\mathcal{D}_t, \mathcal{P}_t) \to (\mathcal{D}_t \cup \{(u_t^\star, y_t)\}, \mathcal{P}_t)$.

2.3 THE SEARCH–EXPERIMENT TRADE-OFF

At each step t, the discovery state is $(\mathcal{D}_t, \mathcal{P}_t)$, and its utility $U(\mathcal{D}_t, \mathcal{P}_t)$. The agent takes an action $a_t \in \{\text{SEARCH}, \text{EXPERIMENT}\}$, and each action is worth the improvement in U it is expected to bring: the *value of computation* (VOC) (Russell & Wefald, 1991) for search, and the *value of information* (VOI) (Howard, 1966) for an experiment at condition u,

$$\begin{aligned} \mathrm{VOC}_t &= \mathbb{E}\big[U(\mathcal{D}_t, \mathcal{P}_{t+1})\big] - U(\mathcal{D}_t, \mathcal{P}_t), \\ \mathrm{VOI}_t(u) &= \mathbb{E}_{Y_u}\big[U(\mathcal{D}_t \cup \{(u, Y_u)\}, \mathcal{P}_t)\big] - U(\mathcal{D}_t, \mathcal{P}_t), \end{aligned} \tag{3}$$

where expectations are over the outcome of the search step and, based on belief, over the outcome Y_u. Neither value can be computed exactly. The two actions consume distinct resources: `Search` uses computation, whereas `Exp` uses laboratory or simulator time, so their values must be weighed against their costs. Let $c(a)$ be the cost of action a, equal to c_S for `Search` and c_E for `Exp`, in the same units as U. Given budgets B^S and B^E, the agent seeks a policy μ that solves:

$$\max_{\mu}\ \mathbb{E}_\mu\Big[U(\mathcal{D}_T, \mathcal{P}_T) - \textstyle\sum_{t<T} c(a_t)\Big] \quad \text{s.t.} \quad N_T^S \le B^S,\ N_T^E \le B^E, \tag{4}$$

where T is the final step and N_T^S and N_T^E count the two actions. The two values are interlinked, as new evidence can make search productive, and better hypotheses can expose more discriminative experiments. The preferred action therefore depends on the current state and on the relative cost $\rho = c_E/c_S$, which makes allocation a sequential, state-dependent decision.

3 COREAGENT FRAMEWORK

At each step t, CoREAgent takes an action $a_t \in \{\texttt{Search}, \texttt{Exp}\}$: `Search` refines the population $\mathcal{P}_t$ on the current evidence, and `Exp` runs the most informative experiment $u_t^\star$ and adds its observation to $\mathcal{D}_t$. The actions draw on separate, non-fungible budgets B^S and B^E and have costs c_S and $c_E(u)$ in units of utility: budgets are hard limits on what a run consumes, while costs are utility penalties that set how much improvement each action must deliver to be preferred (Appendix B). The metareasoner acts on the state $s_t = (\mathcal{D}_t, \mathcal{P}_t, \mathcal{Z}_t, N_t^S, N_t^E)$: the evidence, population, search history, and resources used so far. We instantiate the utility in Eq. 4 as

$$U(\mathcal{D}_t, \mathcal{P}_t) = \mathbb{E}_{u\sim\nu,\ y\sim p^\star(\cdot|u)}\big[\log F_t(y \mid u)\big], \qquad F_t(y \mid u) = \sum_{h\in\mathcal{P}_t} q_t(h)\, p(y \mid h, u), \tag{5}$$

where $p^\star(\cdot \mid u)$ is the true outcome distribution, ν the distribution of conditions on which a mechanism is evaluated, and F_t the population's predictive distribution, which weights each hypothesis's prediction by the belief $q_t(h)$, an approximate posterior that h is the true mechanism. The log score is a proper scoring rule: it is high when high-belief hypotheses predict the true outcomes accurately and with calibrated confidence, so `Search` improves it by changing the hypotheses and `Exp` by changing the evidence that weights them. Since U depends on the unknown system, the metareasoner learns from realized improvements on newly acquired data instead.

3.1 LLM-GUIDED HYPOTHESIS SEARCH

Population-based search. The population $\mathcal{P}_t = \mathcal{P}_t^{\text{explore}} \cup \mathcal{P}_t^{\text{exploit}}$ has two sub-populations, in each of which the LLM proposes hypotheses conditioned on parents from that sub-population: exploration samples at a high temperature for structurally diverse hypotheses, and exploitation at a low temperature to refine promising ones. Each proposal also seeds a symbolic-regression search (Cranmer, 2023); both are fitted to $\mathcal{D}_t$ and kept if valid and novel. To avoid over-parameterized expressions, hypotheses are ranked by their fit $\mathcal{L}_t(h)$ and complexity $C_t(h)$,

$$S_t(h) = \tfrac{1}{2}\Big[\exp\big(-\mathcal{L}_t(h)/\tau\big) + \exp\big(-C_t(h)\big)\Big], \tag{6}$$

where C_t penalizes fitted constants and expression size relative to the data and τ sets the error scale (Appendix B). Both terms lie in $(0, 1]$, and weighted equally. Parents are sampled in favour of well-fitting, compact hypotheses, the top-scoring hypothesis is the current best, and each `Search` step advances both sub-populations once, using two LLM calls.

Cross-population exchange. So that promising exploratory structures are refined while alternatives keep circulating, every few `Search` steps the sub-populations exchange randomly paired hypotheses with the replica-exchange probability of Wang et al. (2026),

$$\alpha = \min\Big\{1,\ \Big(\frac{\mathcal{L}_t(h_{\text{exploit}})}{\mathcal{L}_t(h_{\text{explore}})}\Big)^{\gamma}\Big\}. \tag{7}$$

A swap is always accepted when the exploratory hypothesis fits better, and otherwise becomes less likely the worse it fits; $\gamma > 0$ sets this selectivity and is adapted to keep the acceptance rate near a target (Appendix B).

3.2 HYPOTHESIS-GUIDED EXPERIMENT DESIGN

`Search` yields competing hypotheses that may fit the evidence similarly yet disagree elsewhere. We place a belief q_t on the competitive subset of $\mathcal{P}_t$: one representative per structure, excluding near-duplicates and hypotheses scoring far below the best, which receive zero weight. Under a uniform prior, q_t approximates the posterior with a BIC-style marginal likelihood that rewards fit and penalizes fitted constants (Schwarz, 1978). The next experiment is the one that most reduces the remaining uncertainty per unit cost,

$$I_t(u) = \mathbb{H}[q_t] - \mathbb{E}_{h\sim q_t}\Big[\mathbb{H}\big(q_t(\cdot \mid u, h(u))\big)\Big], \qquad u_t^\star = \arg\max_{u\in\mathcal{U}_t} \frac{I_t(u)}{c_E(u)}, \tag{8}$$

where $\mathcal{U}_t$ is a candidate set of conditions and $q_t(\cdot \mid u, h(u))$ is the belief after observing the outcome h predicts at u, which is exact for our deterministic simulators (Appendix B.2). $I_t(u)$ is the expected

reduction in belief entropy, large where probable hypotheses disagree. An `Exp` step queries the system at $u_t^\star$, adds the observation, refits all hypotheses, and updates q_t. Belief details, the candidate set, and trajectory-valued experiments are given in Appendix B.

3.3 Budget-Aware Search–Experiment Metareasoning

The metareasoner represents s_t by features $\phi_t = \phi(s_t)$ that summarize search progress and diversity, belief uncertainty, the experimental opportunity I_t, and budget usage (Appendix B). Since a run yields only tens of credited decisions, each action has a Bayesian linear value model,

$$Q_a(s_t) = m_a + \phi_t^\top w_a, \qquad w_a \sim \mathcal{N}(\mathbf{0}, \sigma_w^2 I), \tag{9}$$

where $Q_a(s_t)$ estimates the cost-adjusted gain G_t (below) of taking a in s_t and m_a is a prior mean; with Gaussian noise on observed credits, the posterior over w_a is closed-form. At each step, the metareasoner draws $\widetilde{Q}_a(s_t)$ from the posterior of each feasible action and takes $a_t = \arg\max_{a \in \mathcal{A}_t} \widetilde{Q}_a(s_t)$ (Thompson, 1933), where $\mathcal{A}_t$ enforces the budgets and action-specific constraints (Appendix B).

Prequential credit. An action must be judged on evidence it has not seen: `Search` should not be credited for fitting data it already used, nor an `Exp` for predicting its own observation. Following prequential evaluation (Dawid, 1984), let F_t^- be the predictive distribution before action t and F_t^+ the one once its effect is complete: just before the next experiment for `Search`, and right after the new observation for `Exp`. With $\mathcal{T}_t$ the observations from the first experiment after F_t^+ and $\ell(F; \mathcal{T}) = -\frac{1}{|\mathcal{T}|} \sum_{(u,y)\in\mathcal{T}} \log F(y \mid u)$ the mean log loss, the credit is

$$G_t = \ell(F_t^-; \mathcal{T}_t) - \ell(F_t^+; \mathcal{T}_t) - \kappa_t, \tag{10}$$

where κ_t is the cost of the actions between F_t^- and F_t^+. G_t is a myopic, realized proxy for the VOC and VOI of Eq. 3; since experiments target contested conditions rather than ν, it is not an unbiased estimate of the change in U. For `Search`, F_t^+ is the population at the end of its search block, so $Q_{\texttt{Search}}$ estimates the value of continuing to search until the next experiment (worked example in Appendix B.3).

Algorithm 1 summarizes CoREAgent. The LLM first seeds the population from the task description. Each iteration then prepares the decision: it updates the belief over the competitive hypotheses and finds the most informative experiment $u^\star$. The experiment is computed even when it is not run, since its information gain is part of the state the metareasoner observes. The metareasoner draws a value for each feasible action from its posterior and takes the larger, so it mostly follows its current estimates but still tries the less certain action. A `Search` step adds hypotheses to both sub-populations while the data stay fixed; its value is not known yet and is credited later. An `Exp` step queries the oracle at $u^\star$, adds the observation, and refits every hypothesis. Because this observation was unseen by all earlier decisions, it is used to credit the `Search` steps since the last experiment and the previous experiment, and the value posteriors are updated; learning therefore happens only when new evidence arrives. The run ends when a budget is exhausted, or a convergence test on the best hypothesis passes, and the highest-scoring hypothesis is returned.

Algorithm 1 CoREAgent

Require: Data $\mathcal{D}_0$, Oracle $\mathcal{O}$, LLM π_θ, Search budget B^S, Experimental budget B^E
Ensure: Best hypothesis $\hat{h}$

$\mathcal{D} \leftarrow \mathcal{D}_0$; $\mathcal{P} \leftarrow \emptyset$; $N^S, N^E \leftarrow 0$
$w_a \sim \mathcal{N}(0, \sigma_w^2 I)$
while $N^S{+}{+} \le B^S$, $N^E < B^E$, not converged **do**
  $q_t \leftarrow$ belief over $\mathcal{P}$
  $u^\star \leftarrow \arg\max_u I_t(u)/c_E(u)$
  $a_t \leftarrow \arg\max_{a \in \mathcal{A}_t} \widetilde{Q}_a$, $\widetilde{Q}_a \sim p(Q_a \mid s_t)$
  if $a_t =$ `Search` **then**
    $\mathcal{P} \leftarrow \texttt{Search}(\mathcal{P}, \mathcal{D}; \pi_\theta)$
    $N^S \leftarrow N^S{+}{+}$
  else
    $\mathcal{T} \leftarrow \{(u^\star, \mathcal{O}(u^\star))\}$; $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{T}$
    $N^E \leftarrow N^E{+}{+}$
    $G_{t'} \leftarrow$ Eq. 10 on $\mathcal{T}$, $\forall$ pending t'
    refit $\mathcal{P}$ on $\mathcal{D}$; update $p(w_a \mid G)$
  end if
end while
return $\hat{h} = \arg\max_{h \in \mathcal{P}} S_t(h)$

4 EXPERIMENTAL SETUP

Our experiments evaluate both the discovery performance and the allocation behavior of CoREAgent. We focus on the following research questions:

RQ1. Does CoREAgent recover mechanisms more accurately while requiring fewer experiments than state-of-the-art active-discovery methods? **RQ2.** Are the improvements a result of adaptive allocation between search and experimentation, rather than increased search or stronger acquisition alone? **RQ3.** How does CoREAgent adapt its search–experiment allocation as the relative costs of computation and experimentation vary? **RQ4.** What is the contribution of feedback, hypothesis diversity, adaptive allocation, and hypothesis-guided acquisition to overall performance?

Together, these questions assess not only whether CoREAgent improves end-to-end scientific discovery, but also whether these gains arise from its central design principle: adaptively allocating resources between hypothesis search and evidence acquisition. We therefore evaluate across heterogeneous scientific domains, compare against strong task-specific discovery baselines, and analyze how the learned allocation policy contributes to the observed improvements.

Benchmarks. We evaluate CoREAgent on four interactive discovery benchmarks spanning complementary observation regimes and mechanism classes (Appendix A.1). *Point-response benchmarks* return a single observation per experiment. **NewtonBench** (Zheng et al., 2026) evaluates recovery of counterfactual physics laws across 12 domains, while **ActiveSciBench-Chem** (Kabra et al., 2026) requires identifying enzyme-kinetic rate laws from a shared seven-variable experimental interface with hidden variables and parameters. *Trajectory-based benchmarks* instead return an entire dynamical rollout per experiment. **ForceBench** (Murphy, 2026) derived from DiscoverPhysics (Wiemann et al., 2026) contains six two-particle interaction worlds in which controlled probe launches must reveal a non-canonical force law, whereas **ODEBench** (d'Ascoli et al., 2024) comprises dynamical systems queried through initial conditions. All four benchmarks are *memorization-resistant*: the target laws are counterfactual (NewtonBench) or non-canonical (ForceBench), and ActiveSciBench-Chem includes non-textbook mechanisms with hidden inputs and parameters. Following Abhyankar et al. (2026b), we anonymize ODEBench and remove semantic descriptions. Appendix A.1 contains more details about the benchmarks.

Baselines. For each benchmark, we compare CoREAgent against its strongest active-discovery baselines over the same oracle, acquisition budget, and evaluation. We compare CoREAgent with: **(i) Belief-based design** methods maintain an explicit distribution over candidate mechanisms and select informative experiments: Bayesian Experimental Design (BED) uses expected information gain over a specified hypothesis pool, while MDA (Murphy, 2026) maintains a posterior over LLM-proposed models with nested sequential Monte Carlo and selects experiments by expected information gain. **(ii) Search-derived design** methods construct hypotheses through search and use them to guide acquisition; Active PySR builds on PySR (Cranmer, 2023) to select the next experiment at the point of maximum disagreement across the current population. **(iii) LLM-driven discovery** methods couple hypothesis generation with iterative experimentation through fixed control flow, including LLM-AutoSciLab (Kabra et al., 2026), LLM-ACES (Abhyankar et al., 2026b), and DP-Agent (Wiemann et al., 2026). Furthermore, LLM-only alternates between proposing experiments and revising hypotheses with feedback, and LLM+BED replaces its experiment choice with Bayesian experimental design; both follow a fixed Search–Experiment schedule. Additional details are in Appendix A.2.

Evaluation protocol. Every LLM call uses `gpt-4o-mini`. We test CoREAgent with more LLM backbones in Appendix C.3. Each method gets up to $B_E = 20$ experiments beyond a small initial design on NewtonBench, ActiveSciBench-Chem, and ForceBench ($B_E = 10$ on ODEBench; MDA uses its native 10 on ForceBench), and at most $B_S = 400$ LLM calls per task (per state equation on ODEBench). We evaluate each recovered mechanism on held-out data that are never exposed during discovery. For point-response benchmarks and ODEBench, we compute root mean squared logarithmic error (RMSLE) between predicted and true responses, while ForceBench uses normalized mean squared error (NMSE) over held-out trajectories. Following Kabra et al. (2026), we report Exact Accuracy (Acc), where a task is counted as correct if its held-out prediction error falls below a benchmark-specific threshold. We use RMSLE as the error calculation metric across NewtonBench, ActiveSciBench-Chem, and ODEBench, having ForceBench evaluated using the normalized error. We additionally report the number of experiments consumed before termination. Appendix A.3 provides more details about the evaluation metrics including the benchmark-specific thresholds.

5 RESULTS & DISCUSSION

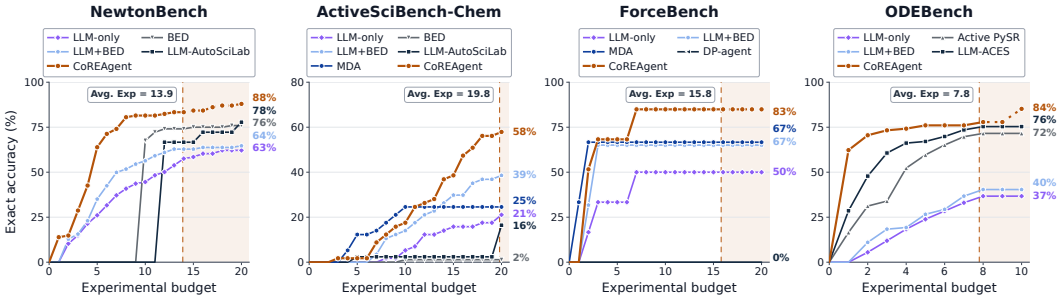


Figure 2: **Performance versus experimental cost.** CoREAgent achieves higher exact accuracy than other baselines with fewer experimental acquisitions. Dashed lines indicate the mean stopping point for CoREAgent.

5.1 RQ1: DISCOVERY PERFORMANCE AND EXPERIMENTAL EFFICIENCY

To quantify CoREAgent's performance improvements across datasets and baselines, we evaluate recovery as a function of the experimental budget consumed. As shown in Figure 2, the central advantage of CoREAgent is that *it achieves stronger mechanism recovery while requiring fewer experimental acquisitions*. The dashed vertical line marks the mean experimental budget at which CoREAgent stops after detecting convergence, showing that the framework typically reaches its final recovery well before the maximum acquisition budget is exhausted. On NewtonBench, CoREAgent improves performance by 10 percentage points over the strongest baseline, LLM-AutoSciLab, while using only 13.9 experiments on average (median 13), corresponding to a 31% reduction in experimental budget. On ODEBench, CoREAgent surpasses the full-budget performance of LLM-ACES and Active PySR while using only 7.8 experiments on average (median 6) with consistent performance across 1D, 2D, and 3D systems, with dimension-stratified results reported in Appendix C. On ForceBench, DP-Agent's held-out error exceeds the threshold on all tasks (0% Acc), while CoREAgent follows the same efficiency trend, reaching 83% recovery with only 15.8 experiments on average while outperforming methods that continue to use the full acquisition budget. ActiveSciBench-Chem is inherently more difficult, and early convergence occurs on only about 10% of tasks, so reductions in experimental usage are limited. Even so, CoREAgent achieves 58% recovery, nearly a 20 percentage-point improvement over the strongest baseline at 39%. MDA operates over benchmark-specific, pre-specified design libraries that substantially narrow the experiment space; on ForceBench, for example, it selects from only 13 fixed designs and runs for 10 experiments. Despite this prior structuring, CoREAgent achieves substantially higher mechanism recovery, demonstrating stronger experimental efficiency without requiring a benchmark-specific shortlist. Overall, CoREAgent *extracts substantially more mechanism recovery per experiment, reaching stronger solutions earlier and with markedly fewer acquisitions across most benchmarks.*

5.2 RQ2: PERFORMANCE GAINS FROM ADAPTIVE ALLOCATION

We further investigate whether the gains of CoREAgent are driven by greater hypothesis-search expenditure, or due to adaptive allocation between search and experimentation. To assess this, all methods are given the same search and experimental budgets. However, due to the missing metareasoner, LLM-only and LLM+BED run until budget exhaustion. Results from Table 1 show that CoREAgent achieves substantially higher recovery while consuming less of both resources. On NewtonBench, it reaches 87.9% accuracy using only 40.3% of the search budget and 69.5% of the experimental budget. On ODEBench, CoREAgent attains 83.6% exact accuracy while using just 43.3% and 78.7% of the search and experiment budgets, respectively.

Table 1: **Higher accuracy under lower resource consumption.** All methods receive identical maximum search and experimental budgets; reported percentages indicate the fraction of each budget actually consumed.

Method	Acc. ↑	Search (%) ↓	Exp. (%) ↓
NewtonBench			
LLM-only	63.5	100.0	100.0
LLM+BED	63.5	100.0	100.0
CoREAgent	**87.9**	**40.3**	**69.5**
ODEBench			
LLM-only	36.7	100.0	100.0
LLM+BED	40.4	100.0	100.0
CoREAgent	**83.6**	**43.3**	**78.7**

Crucially, CoREAgent achieves these gains while using substantially less search, showing that improved recovery is not a consequence of additional hypothesis-search computation. Moreover, replacing the fixed experiment policy with BED yields limited improvement, indicating that stronger acquisition alone is also insufficient. The gains instead reflect the full CoREAgent pipeline, consisting of hypothesis search that produces diverse mechanisms, experiment design that maximizes information gain, and adaptive selection between the two with evolving marginal values.

5.3 RQ3: Cost-Sensitive Resource Allocation

To test whether CoREAgent responds to the relative cost of search and experimentation, we exploit the simulator setting to vary one action cost at a time while keeping the other cost, tight budgets (B^S=400, B^E=20), and acquisition policy fixed. This enables a controlled sweep across cost regimes; Figure 3 reports the resulting changes over 15 NewtonBench tasks.

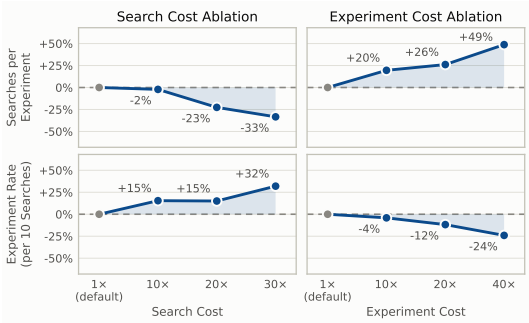


Figure 3: **Cost-sensitive resource allocation.** Increasing search cost reduces search frequency and increases experimentation, and vice versa.

Sensitivity to search cost. As search becomes more expensive, CoREAgent reduces refinement and shifts effort toward experimentation. Searches per experiment fall by 23% and 33% at 20× and 30× cost, while the experiment rate rises by up to 32%. This reallocation is driven by the metareasoner's action choices rather than budget exhaustion, and recovery is preserved, with accuracy remaining at 80% even at 30× search cost. Thus, the gains of CoREAgent do not rely on simply performing more search.

Sensitivity to experiment cost. Increasing experiment cost induces the complementary behavior. Searches per experiment rise by 20%, 26%, and 49%, while the experiment rate falls by up to 24%, showing that CoREAgent extracts more value from existing evidence before acquiring new observations. Performance remains close to the default setting, confirming that the metareasoner adaptively reallocates effort as the relative cost of the two actions changes. Analyses on generalization across LLMs, per-dimension performance, and computation are in Appendix C.

5.4 RQ4: Ablation Study

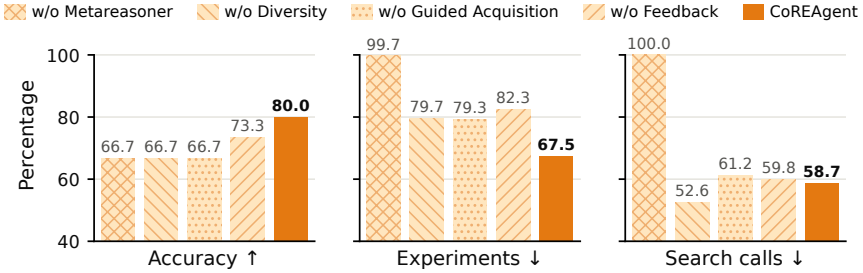


Figure 4: **Ablation study.** Accuracy and resource consumption over 15 NewtonBench tasks under identical budgets. Removing any component reduces accuracy and generally increases resource consumption.

We study the contribution of four components of CoREAgent on 15 NewtonBench tasks under identical search and experimental budgets. **'*w/o* Feedback'** removes feedback from newly acquired evidence during hypothesis refinement. **'*w/o* Diversity'** removes the exploratory mechanism, reducing explicit preservation of structurally distinct hypotheses. **'*w/o* Metareasoner'** replaces adaptive Search–Experiment allocation with a fixed schedule until budget exhaustion. **'*w/o* Guided Acquisition'** replaces hypothesis-conditioned experiment selection with uniform sampling over the feasible design space. As shown in Figure 4, the full CoREAgent achieves the highest accuracy (80.0%) while using only 67.5% of the experimental budget and 58.7% of the search budget. Removing the metareasoner consumes 1.5× the experiments and 1.7× the search (99.7% and 100% of the budgets) yet lowers accuracy to 66.7%, highlighting that the performance gains cannot be attributed to additional computation or experimentation. Removing guided acquisition also lowers accuracy to 66.7% despite increasing experiment usage to 79.3%, highlighting the importance of selecting

experiments that discriminate among competing hypotheses. Without diversity, search usage falls to 52.6%, but accuracy drops to 66.7%, indicating that reduced search is not beneficial when it comes at the cost of prematurely narrowing the hypothesis space. Finally, removing feedback yields 73.3% accuracy while increasing experiment usage to 82.3%. Every ablation lowers accuracy by 6.7–13.3 points, and none recovers it even with more budget, so CoREAgent's gains require adaptive allocation, hypothesis diversity, guided acquisition, and closed-loop feedback together.

6 Related Work

LLM-guided Scientific Discovery. LLMs are increasingly used across the scientific workflow for literature synthesis, hypothesis generation, experimental planning, and data-driven discovery (AI4Science & Quantum, 2023; Reddy & Shojaee, 2025; Asai et al., 2026; Jansen et al., 2026; Majumder et al., 2025). A growing line of work further casts discovery as iterative hypothesis search, where candidates are proposed, evaluated, and refined, spanning mathematical programs (Romera-Paredes et al., 2024), symbolic regression (Grayeli et al., 2024; Shojaee et al., 2025), molecular and materials discovery (Wang et al., 2025b; Ojeda et al., 2026; Abhyankar et al., 2026a), protein optimization (Wang et al., 2025c), and open-ended search (Agarwal et al., 2025). However, better search cannot resolve limitations of the available evidence, as distinct hypotheses may explain the same observations while making different predictions elsewhere. Resolving such identifiability gaps requires actively acquiring observations that distinguish competing explanations.

Experimental Design for Active Discovery. Adaptive experimentation spans materials and process discovery (Ling et al., 2017; Kusne et al., 2020), molecular and drug discovery (Bailey et al., 2023; Kyro et al., 2024), genomics (Huang et al., 2023; Qin et al., 2024), and physics (Melnikov et al., 2018). Recent LLM systems close the hypothesis–experiment loop through perturbation design (Roohani et al., 2025), hypothesis revision with informative experiments (Piriyakulkij et al., 2024), and active discrimination among competing theories (Abhyankar et al., 2026b; Éltető et al., 2026; Prystawski et al., 2026; Jagadish et al., 2026). More recent methods combine disagreement-driven acquisition (Kabra et al., 2026), bayesian inference and value-of-information design (Murphy, 2026), or sequential experimentation benchmarks (Wiemann et al., 2026; Zheng et al., 2026). These approaches improve experiment selection under limited query budgets, but generally organize discovery around repeated data acquisition rather than explicitly comparing the value of further hypothesis search on existing evidence against the higher cost of obtaining new evidence. This *when-to-experiment* decision motivates a budget-aware treatment of scientific discovery.

Budget-aware LLM systems. This has become increasingly relevant for LLMs, motivating cost-aware model cascades and routing (Chen et al., 2024; Aggarwal et al., 2024; Ong et al., 2025), adaptive reasoning compute (Han et al., 2025; Wen et al., 2025; Li et al., 2026; Wang et al., 2025a), and budget-aware agents, tool use, and resource estimation (Liu et al., 2026; Lin et al., 2026; Zhang et al., 2026a). A related line learns lightweight value estimators for model selection or early termination (Zhang et al., 2026b; Davidov et al., 2026). These works primarily study how computation, models, or tools should be allocated within a fixed external task. CoREAgent instead adaptively selects between asymmetric-cost actions: `Search` to refine hypotheses from existing evidence and `Experiment` to acquire new evidence using a lightweight metareasoner that estimates their downstream value.

7 Conclusion

We introduce CoREAgent, a budget-aware metareasoning framework for scientific discovery that explicitly decides *when to refine hypotheses* and *when to acquire new evidence* under separate computational and experimental budgets. CoREAgent makes Search–Experiment allocation itself a learned decision problem where the metareasoner estimates the state-dependent, cost-adjusted value of each action and learns these values online through prequential feedback from future evidence. Across four interactive discovery benchmarks, CoREAgent consistently improves performance while using resources more efficiently. More broadly, our results suggest that scientific discovery should be viewed not as a hypothesis refinement or experiment selection, but as a resource-allocation problem in which the value of computation and experimentation changes throughout the discovery process. This perspective elevates *when to search and when to experiment* to a first-class decision, complementing prior work on what hypotheses to explore and where to acquire evidence.

Limitations. Our current evaluation focuses on simulator-based, largely deterministic mechanism-discovery settings with structured hypothesis spaces. Extending this setting to broader mechanism classes and real laboratory environments with noise, delays, safety constraints, and heterogeneous costs remains an important direction for future extensions.

AI USE STATEMENT

This work studies LLM-driven scientific discovery, and LLMs are a component of both CoREAgent, where they propose and refine candidate hypotheses, and the evaluated baselines. Sections 3 and 4 mention the details of the LLM used in our proposed framework and baselines. The authors also used generative AI tools during implementation and manuscript preparation, for code assistance and debugging, drafting support, and editing suggestions. The authors checked all experimental results, claims, tables, and figures and take responsibility for the final content of the submission.

ETHICS STATEMENT

This work uses publicly available scientific discovery benchmarks whose experiments are executed by simulators; it does not involve human subjects, personal data, or experiments on physical systems. CoREAgent is a research framework, not an autonomous controller for real laboratories. Because its metareasoner decides when new experiments are run, deploying it with physical experiments should add domain-specific feasibility and safety constraints on the experiments it may request, and human approval before any experiment is executed.

REPRODUCIBILITY STATEMENT

We evaluate in simulator-based environments rather than physical laboratories to enable fully reproducible closed-loop discovery and controlled variation of experimental cost. We provide an anonymous code repository[2] and detailed specifications of the experimental pipeline. Section 3, Algorithm 1, and Appendix B describe CoREAgent, in detail. Section 4 specifies the evaluation protocol and the search and experiment budgets. Appendix A.1 documents the benchmarks, experimental interfaces, and held-out evaluation, while Appendix A.3 details the evaluation metrics. Additional implementation and reproducibility details are provided in the accompanying code repository.

REFERENCES

Nikhil Abhyankar, Sanchit Kabra, Saaketh Desai, and Chandan K. Reddy. LLEMA: Evolutionary search with LLMs for multi-objective materials discovery. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=TIqzhBvCNB.

Nikhil Abhyankar, Sha Li, Sanchit Kabra, Naren Ramakrishnan, Yulia Gel, and Chandan K Reddy. Llm-aces: Closed-loop discovery of dynamical systems with llm-guided adaptive search. *arXiv preprint arXiv:2606.25039*, 2026b.

Dhruv Agarwal, Bodhisattwa Prasad Majumder, Reece Adamson, Megha Chakravorty, Satvika Reddy Gavireddy, Aditya Parashar, Harshit Surana, Bhavana Dalvi Mishra, Andrew McCallum, Ashish Sabharwal, and Peter Clark. AutoDiscovery: Open-ended scientific discovery via bayesian surprise. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0847.

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Manaal Faruqui, and Mausam. AutoMix: Automatically mixing language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4164.

Microsoft Research AI4Science and Microsoft Azure Quantum. The impact of large language models on scientific discovery: a preliminary study using gpt-4. *arXiv preprint arXiv:2311.07361*, 2023.

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'arcy, et al. Synthesizing scientific literature with retrieval-augmented language models. *Nature*, 650(8103):857–863, 2026.

[2] https://anonymous.4open.science/r/core_agent

Michael Bailey, Saeed Moayedpour, Ruijiang Li, Alejandro Corrochano-Navarro, Alexander Kötter, Lorenzo Kogler-Anele, Saleh Riahi, Christoph Grebner, Gerhard Hessler, Hans Matter, et al. Deep batch active learning for drug discovery. *bioRxiv*, pp. 2023–07, 2023.

George E. P. Box and William J. Hill. Discrimination among mechanistic models. *Technometrics*, 9 (1):57–71, 1967. doi: 10.1080/00401706.1967.10490441.

Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical science*, pp. 273–304, 1995.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024.

Miles Cranmer. Interpretable machine learning for science with pysr and symbolicregression. jl. *arXiv preprint arXiv:2305.01582*, 2023.

Stéphane d'Ascoli, Sören Becker, Philippe Schwaller, Alexander Mathis, and Niki Kilbertus. Odeformer: Symbolic regression of dynamical systems with transformers. In *International Conference on Learning Representations*, volume 2024, pp. 21943–21976, 2024.

Hen Davidov, Nachshon Cohen, Oren Kalinsky, Yaron Fairstein, Guy Kushilevitz, Ram Yazdi, and Patrick Rebeschini. Knowing when to quit: A principled framework for dynamic abstention in LLM reasoning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

A. Philip Dawid. Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society: Series A (General)*, 147 (2):278–292, 1984.

Noémi Éltető, Nathaniel D Daw, Kimberly L Stachenfeld, and Kevin J Miller. Atlas: Active theory learning for automated science. *arXiv preprint arXiv:2606.12386*, 2026.

Arya Grayeli, Atharva Sehgal, Omar Costilla-Reyes, Miles Cranmer, and Swarat Chaudhuri. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37: 44678–44709, 2024.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.1274.

John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, 1966. doi: 10.1109/TSSC.1966.300074.

Kexin Huang, Romain Lopez, Jan-Christian Hütter, Takamasa Kudo, Antonio Rios, and Aviv Regev. Sequential optimal experimental design of perturbation screens guided by multi-modal priors. *bioRxiv*, 2023. doi: 10.1101/2023.12.12.571389.

Akshay K Jagadish, Younes Strittmatter, Nori Jacoby, George Kachergis, Eric Schulz, Nathaniel Daw, Suyog H Chandramouli, and Thomas L Griffiths. Closing the loop to discover psychological theories with an automated cognitive scientist. *arXiv preprint arXiv:2606.26448*, 2026.

Peter Jansen, Peter Clark, Doug Downey, and Daniel S Weld. Generating literature-driven scientific theories at scale. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14677–14698, 2026.

Sanchit Kabra, Nikhil Abhyankar, Saaketh Desai, Prasad Iyer, and Chandan K Reddy. Llm-autoscilab: Closed-loop scientific discovery via active experimentation with llms. *arXiv preprint arXiv:2605.24043*, 2026.

A. Gilad Kusne, Heshan Yu, Changming Wu, Huairuo Zhang, Jason Hattrick-Simpers, Brian DeCost, Suchismita Sarker, Corey Oses, Cormac Toher, Stefano Curtarolo, Albert V. Davydov, Ritesh Agarwal, Leonid A. Bendersky, Mo Li, Apurva Mehta, and Ichiro Takeuchi. On-the-fly closed-loop materials discovery via bayesian active learning. *Nature Communications*, 11:5966, 2020. doi: 10.1038/s41467-020-19597-w.

Gregory W. Kyro, Anton Morgunov, Rafael I. Brent, and Victor S. Batista. ChemSpaceAL: An efficient active learning methodology applied to protein-specific molecular generation. *Journal of Chemical Information and Modeling*, 64(3):653–665, 2024. doi: 10.1021/acs.jcim.3c01456.

Zheng Li, Qingxiu Dong, Jingyuan Ma, Di Zhang, Kai Jia, and Zhifang Sui. SelfBudgeter: Adaptive token allocation for efficient LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 21135–21156. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1063.

Yuxiang Lin, Zihan Wang, Mengyang Liu, Yuxuan Shan, Longju Bai, Junyao Zhang, Xing Jin, Boshan Chen, Jinyan Su, Xingyao Wang, et al. Bagen: Are llm agents budget-aware? *arXiv preprint arXiv:2606.00198*, 2026.

Dennis V. Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956. doi: 10.1214/aoms/1177728069.

Julia Ling, Maxwell Hutchinson, Erin Antono, Sean Paradiso, and Bryce Meredig. High-dimensional materials and process optimization using data-driven experimental design with well-calibrated uncertainty estimates. *Integrating Materials and Manufacturing Innovation*, 6(3):207–217, 2017.

Tengxiao Liu, Zifeng Wang, Jin Miao, I-Hung Hsu, Jun Yan, Jiefeng Chen, Rujun Han, Fangyuan Xu, Yanfei Chen, Ke Jiang, Samira Daruki, Yi Liang, William Yang Wang, Tomas Pfister, and Chen-Yu Lee. Budget-aware tool-use enables effective agent scaling. In *Third Conference on Language Modeling*, 2026.

Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Bhavana Dalvi Mishra, Abhijeetsingh Meena, Aryan Prakhar, Tirth Vora, Tushar Khot, Ashish Sabharwal, and Peter Clark. DiscoveryBench: Towards data-driven discovery with large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Alexey A. Melnikov, Hendrik Poulsen Nautrup, Mario Krenn, Vedran Dunjko, Markus Tiersch, Anton Zeilinger, and Hans J. Briegel. Active learning machine learns to create new quantum experiments. *Proceedings of the National Academy of Sciences*, 115(6):1221–1226, 2018. doi: 10.1073/pnas.1714936115.

Kevin Murphy. Model discovery agent: Llm-assisted bayesian experiment design for data-efficient discovery of mechanistic world models. *arXiv preprint arXiv:2608.09696*, 2026.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

César Ojeda, Darius A Faroughy, Maryam Karimi, Payam Zarrintaj, Mir Mehdi Seyedebrahimi, and Martín Carballo-Pacheco. My chemical harness: Evolutionary molecular design over synthetic pathways with large language model agents. *arXiv preprint arXiv:2606.11256*, 2026.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wasu Top Piriyakulkij, Cassidy Langenfeld, Tuan Anh Le, and Kevin Ellis. Doing experiments and revising rules with natural language and probabilistic reasoning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 53102–53137, 2024. doi: 10.52202/079017-1682.

Ben Prystawski, Kushin Mukherjee, Daniel Wurgaft, Linas Nasvytis, Michael Y Li, Noah D Goodman, and Michael C Frank. auto-psych: Automating the science of mind using agent-driven theory discovery and experimentation. *arXiv preprint arXiv:2606.26460*, 2026.

Jason Qin, Hans-Hermann Wessels, Carlos Fernandez-Granda, and Yuhan Hao. Active learning for efficient discovery of optimal gene combinations in the combinatorial perturbation space. In *NeurIPS 2024 Workshop on AI for New Drug Modalities*, 2024. URL https://openreview.net/forum?id=7aCRpxvu2N.

Tom Rainforth, Adam Foster, Desi R. Ivanova, and Freddie Bickford Smith. Modern bayesian experimental design. *Statistical Science*, 39(1):100–114, 2024. doi: 10.1214/23-STS915.

Ingo Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, volume 15 of *Problemata*. Frommann-Holzboog, Stuttgart-Bad Cannstatt, 1973.

Chandan K Reddy and Parshin Shojaee. Towards scientific discovery with generative AI: Progress, opportunities, and challenges. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 28601–28609, 2025.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

Yusuf Roohani, Andrew Lee, Qian Huang, Jian Vora, Zachary Steinhart, Kexin Huang, Alexander Marson, Percy Liang, and Jure Leskovec. BioDiscoveryAgent: An AI agent for designing genetic perturbation experiments. In *The Thirteenth International Conference on Learning Representations*, 2025.

Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1–3):361–395, 1991. doi: 10.1016/0004-3702(91)90015-C.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

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*, 2025. URL https://openreview.net/forum?id=m2nmp8P5in.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

Gary Tom, Stefan P Schmid, Sterling G Baird, Yang Cao, Kourosh Darvish, Han Hao, Stanley Lo, Sergio Pablo-García, Ella M Rajaonson, Marta Skreta, et al. Self-driving laboratories for chemistry and materials science. *Chemical Reviews*, 124(16):9633–9732, 2024.

Fali Wang, Hui Liu, Zhenwei Dai, Jingying Zeng, Zhiwei Zhang, Zongyu Wu, Chen Luo, Zhen Li, Xianfeng Tang, Qi He, and Suhang Wang. AgentTTS: Large language model agent for test-time compute-optimal scaling strategy in complex tasks. In *Advances in Neural Information Processing Systems*, volume 38, pp. 109051–109088, 2025a. doi: 10.52202/085713-3288.

Fiona Y Wang and Markus J Buehler. Self-revising discovery systems for science: A categorical framework for agentic artificial intelligence. *arXiv preprint arXiv:2606.01444*, 2026.

Haorui Wang, Marta Skreta, Cher Tian Ser, Wenhao Gao, Lingkai Kong, Felix Strieth-Kalthoff, Chenru Duan, Yuchen Zhuang, Yue Yu, Yanqiao Zhu, Yuanqi Du, Alan Aspuru-Guzik, Kirill Neklyudov, and Chao Zhang. Efficient evolutionary search over chemical space with large language models. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=awWiNvQwf3.

Haorui Wang, Parshin Shojaee, Kazem Meidani, Kunyang Sun, José Miguel Hernández-Lobato, Teresa Head-Gordon, Jiajun He, Chandan K. Reddy, Chao Zhang, and Yuanqi Du. Towards diverse scientific hypothesis search with large language models. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

Yinkai Wang, Jiaxing He, Yuanqi Du, Xiaohui Chen, Jianan Canal Li, Li-Ping Liu, Xiaolin Xu, and Soha Hassoun. Large language model is secretly a protein sequence optimizer. *arXiv preprint arXiv:2501.09274*, 2025c.

Hao Wen, Xinrui Wu, Yi Sun, Feifei Zhang, Liye Chen, Jie Wang, Yunxin Liu, Yunhao Liu, Ya-Qin Zhang, and Yuanchun Li. Budgetthinker: Empowering budget-aware llm reasoning with control tokens. *arXiv preprint arXiv:2508.17196*, 2025.

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.

Shijie Xia, Yuhan Sun, and Pengfei Liu. Sr-scientist: Scientific equation discovery with agentic ai. In *International Conference on Learning Representations*, volume 2026, pp. 75787–75811, 2026.

Jieyu Zhang, Ranjay Krishna, Ahmed H Awadallah, and Chi Wang. Ecoassistant: Using llm assistant more affordably and accurately. *arXiv preprint arXiv:2310.03046*, 2023.

Yansen Zhang, Yilu Liu, Tianyu Liu, Jiamin Chen, Xiaokun Zhang, Kai Xie, Xue Liu, Yiyan Qi, and Chen Ma. Budget-aware llm discovery via cost-calibrated frontier utility. *arXiv preprint arXiv:2607.26828*, 2026a.

Yiqun Zhang, Hao Li, Zihan Wang, Shi Feng, Xiaocui Yang, Daling Wang, Bo Zhang, Lei Bai, and Shuyue Hu. MTRouter: Cost-aware multi-turn LLM routing with history–model joint embeddings. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44206–44226. Association for Computational Linguistics, 2026b. doi: 10.18653/v1/2026.acl-long.2045.

Tianshi Zheng, Zheye Deng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Zihao Wang, and Yangqiu Song. From automation to autonomy: A survey on large language models in scientific discovery. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17733–17750. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.895.

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.

APPENDIX

A IMPLEMENTATION DETAILS

A.1 DATASETS

Table 2 summarizes the four interactive discovery benchmarks used in our evaluation. In each benchmark, the agent queries the experimental interface and is evaluated on held-out conditions that it never queries during discovery. Every run starts from a single seed experiment, after which the agent chooses at most B^E further experiments.

Table 2: Benchmark and evaluation summary.

Benchmark	Target	Tasks	Experiment output	B^E	Held-out evaluation
NewtonBench	Counterfactual physical law	108	Scalar response at queried inputs	20	1000 fresh input assignments
ActiveSciBench-Chem	Enzyme-kinetic mechanism	57	Initial rate at a 7-D assay condition	20	1000 fresh assay conditions
ForceBench	Non-canonical force law	6	Probe trajectory from a selected launch	20	6 launches with unseen coupling knobs
ODEBench	Dynamical system	61	Trajectory on from a selected initial condition	10	5 unseen initial conditions over $t \in [0, 10]$

NewtonBench. NewtonBench (Zheng et al., 2026) contains counterfactual variants of physical laws spanning 12 domains and multiple levels of system complexity. Following Kabra et al. (2026), we evaluate the 108 vanilla-equation tasks across 12 domains (gravitation, Coulomb force, magnetic force, Fourier's law, Snell's law, radioactive decay, underdamped oscillation, Malus's law, speed of sound, Hooke's law, Bose–Einstein distribution, and heat transfer) $\times$ three equation difficulties (easy, medium, hard) $\times$ three counterfactual versions of each law. An experiment specifies values for the law's input variables within fixed per-domain ranges and returns the scalar response of the hidden law. The held-out set contains 1000 input assignments drawn log-uniformly from the same ranges, and a law counts as recovered when its RMSLE against the noise-free response is below 0.01. We use the simulator modules of their official implementation (`HKUST-KnowComp/NewtonBench`), with the input ranges and held-out scoring given by LLM-AutoSciLab.

ActiveSciBench-Chem. ActiveSciBench-Chem (Kabra et al., 2026), comprises 57 enzyme-kinetic mechanisms of varying difficulty, from single mechanisms such as Michaelis–Menten kinetics to compound mechanisms that combine saturation, inhibition, cooperativity, bisubstrate, and temperature effects. Each experiment specifies substrate ($C_A \in [0.01, 100]$ mM), inhibitor ($C_I \in [0, 50]$ mM), secondary-substrate ($C_B \in [0.01, 100]$ mM), product ($C_P \in [0, 20]$ mM), enzyme ($[0.01, 10]$ mg/mL), temperature ($T \in [278, 368]$ K), and pH ($[4, 10]$) conditions and returns the resulting initial reaction rate. All seven inputs are exposed, while the relevant variables, functional form, and parameters remain hidden. The held-out set contains 1000 assay conditions sampled over the full input ranges. We use the released LLM-AutoSciLab implementation (`scientific-discovery/LLM-AutoSciLab`).

ForceBench. ForceBench is derived from DiscoverPhysics (Wiemann et al., 2026). It is the wrapper introduced with the Model Discovery Agent (MDA) (Murphy, 2026), which leaves the DiscoverPhysics simulator unchanged and only replaces its interface to match the other benchmarks. We use the six two-particle worlds, as shown in Table 3. An experiment specifies a probe launch and returns its resulting trajectory. A launch sets the release radius r_0, a tangential speed v_t, and two coupling knobs (p_1, p_2) whose physical roles (source charge or probe inertia) are themselves part of what must be discovered. Trajectories are simulated with the DiscoverPhysics n-body engine. We follow the evaluation protocol of MDA (Murphy, 2026): held-out launches use more extreme knob settings than any training launch, and a law passes when its normalized MSE on the noise-free held-out trajectories is at most 0.1. The underlying simulator is available in the DiscoverPhysics repository (`SampsonML/DiscoverPhysics`).

Table 3: **ForceBench worlds and settings.** Top: the six two-particle worlds, with force $k\, q_i q_j\, F(r, t)$ along the separation vector; the key is the world's name in the DiscoverPhysics registry. Bottom: experiment and evaluation settings.

World	DiscoverPhysics key	$F(r, t)$	Notes
Gravity	`gravity`	$1/r$	Inverse-distance law
Yukawa	`yukawa`	$K_1(r/\lambda)/\lambda$, $\lambda = 2$	$\approx 1/r$ for $r \ll \lambda$, exponentially screened beyond λ
Coulomb	`coulomb_easy`	$1/r^2$	Inverse-square law
Oscillator	`oscillator`	$\cos(\omega t + \phi)/r$	Coupling periodically reverses sign
Fractional	`fractional`	$1/r^{3-2\alpha}$, $\alpha = \frac{1}{2}$	Fractional Laplacian; equals $1/r^2$
Extra dimensions	`extra_dimensions`	$\Phi_{\mathrm{KK}}(r)$	Kaluza–Klein kernel: $1/r^2$ at short range, $1/r$ at long range
Training launches	256-point scrambled Sobol pool over $r_0 \in [1.5, 10]$, $v_t \in [0, 0.4]$, $p_1, p_2 \in [1, 2]$; read at $t \in \{0.5, 1, 1.5, 2, 3, 4\}$		
Held-out launches	$(p_1, p_2) \in \{3, 4, 5\} \times \{3, 5\}$, paired with radial drops from $r_0 = 3$ and $r_0 = 6$ and a tangential launch from $r_0 = 4$ with $v_t = 0.4$; read at $t = 1, \ldots, 10$		
Simulator	n-body engine, fourth-order Yoshida integrator, $\Delta t = 0.005$, softening 0.05		
Pass criterion	nMSE ≤ 0.1 on noise-free held-out trajectories		

ODEBench. ODEBench (d'Ascoli et al., 2024) contains 63 dynamical systems, collected from Strogatz's textbook, spanning one to four state variables; each system comes with fixed constants and two prescribed initial conditions. We use its 61 1-D, 2-D and 3-D systems. In our setting, an experiment selects an initial condition within the per-system box used by LLM-ACES (Abhyankar et al., 2026b) and returns the corresponding numerical trajectory on $t \in [0, 1]$ at 20 evenly spaced times, together with the time derivatives at those states. All trajectories are integrated with ODEBench's

solver settings (LSODA, $\mathrm{rtol} = 10^{-5}$, $\mathrm{atol} = 10^{-7}$). We evaluate on five unseen held-out initial conditions over the full window $t \in [0, 10]$. We use the standalone ODEBench implementation (`GPBench/ODEBench`) together with the data splits and initial-condition boxes of LLM-ACES (`scientific-discovery/LLM-ACES`).

A.2 BASELINES

We evaluate all methods through the same benchmark interfaces and held-out evaluation protocol described in Section 4. Unless otherwise specified, all LLM-based methods use `gpt-4o-mini`. We allow at most 20 experimental acquisitions on NewtonBench, ActiveSciBench-Chem, and ForceBench, and 10 initial-condition queries on ODEBench. Existing methods are run from their released implementations when available or implemented following their published algorithms.

LLM-only. We implement an iterative LLM-based symbolic discovery baseline following the LLM-only protocol of NewtonBench (Zheng et al., 2026) and the iterative refinement setting used by LLM-ACES (Abhyankar et al., 2026b). At each round, the LLM receives the task description, accumulated observations, and previously discovered high-scoring hypotheses, and proposes new candidate equations. Candidate constants are fitted to all available evidence and the best hypotheses are retained as feedback for subsequent proposals. Search follows a fixed schedule, and new experiments are sampled uniformly from the feasible design space. This baseline captures LLM-guided hypothesis refinement without adaptive experiment selection or search–experiment allocation.

LLM+BED. LLM+BED uses exactly the same hypothesis-generation procedure and fixed search schedule as LLM-only, but replaces uniform acquisition with Bayesian experimental design. A belief is formed over the current competitive hypotheses, and the next experiment maximizes expected information gain about their identity. This provides a controlled comparison in which hypothesis search is unchanged and only the acquisition rule is strengthened.

Bayesian Experimental Design (BED). We adapt the BED baseline from LLM-AutoSciLab (Kabra et al., 2026). BED maintains a fixed library of parametric mechanism families, fits each candidate to the observations collected so far, and selects experiments where the fitted mechanisms exhibit the greatest predictive disagreement. Unlike the LLM-based methods, its hypothesis space is fixed throughout discovery. For domains where the released mechanism library is not applicable, we use a generic symbolic library of matched size; no subsequent LLM or symbolic-search refinement is performed.

Model Discovery Agent (MDA). MDA (Murphy, 2026) combines LLM-based mechanism proposals with Bayesian inference and value-of-information experiment design. Candidate structures proposed by the LLM are assigned posterior probabilities using sequential Monte Carlo, and experiments are selected to maximally reduce uncertainty over the candidate mechanisms. Predictive failures can trigger additional hypothesis proposals, allowing the hypothesis set to expand during discovery. Since the released repository[3] does not provide an executable implementation of the complete method, we reimplemented MDA following its published algorithms and use the reported benchmark-specific settings where applicable.

Active PySR. Active PySR is the query-by-committee symbolic-regression baseline used by LLM-ACES (Abhyankar et al., 2026b). PySR (Cranmer, 2023) fits a set of candidate dynamical equations to the trajectories acquired so far, from which a committee of competing models is constructed. The next initial condition is chosen to maximize disagreement among their predicted trajectories, after which PySR is refit on the expanded dataset. This provides an active-discovery baseline without LLM-guided hypothesis generation.

LLM-ACES. LLM-ACES (Abhyankar et al., 2026b) couples LLM-guided symbolic hypothesis construction with active trajectory acquisition. At each round, the LLM proposes structured operator priors, PySR searches within the corresponding symbolic subspaces, and predictive disagreement

[3] `https://github.com/murphyk/neuronbench`

among the resulting equations guides selection of the next initial condition. Newly acquired trajectories are then used to rescore hypotheses and update an experience buffer that conditions subsequent proposals. We run the released implementation [4] using the common LLM backbone and acquisition budget.

LLM-AutoSciLab. LLM-AutoSciLab (Kabra et al., 2026) is a closed-loop scientific discovery agent that alternates between hypothesis generation, experimental design, and feedback-driven refinement. It maintains positive and negative evidence from previous hypotheses and uses this information to guide subsequent proposals. Its acquisition policy adaptively selects experimental regions using uncertainty- and improvement-based criteria, while candidate mechanisms are refined through numerical fitting and symbolic regression. We use the released implementation [5] with its default discovery policy, replacing the original LLM backbone with the common backbone used throughout our evaluation.

DiscoverPhysics Agent (DP-Agent). DP-Agent[6] is the reference discovery agent from DiscoverPhysics (Wiemann et al., 2026). The LLM iteratively proposes interventions, analyzes observed trajectories, optionally fits parameters of candidate mechanisms, and ultimately returns an executable force law. To make experimental consumption directly comparable with the other methods, we evaluate the agent through the same ForceBench experiment interface and acquisition budget used throughout our study.

A.3 Evaluation Metrics

For each task τ in a benchmark $\mathcal{T}$, let $\hat{h}_\tau$ be the mechanism returned at termination and $\mathcal{D}_\tau^{\text{test}}$ its held-out evaluation set, which the agent never queried. Following Kabra et al. (2026), a task counts as exactly recovered if the held-out error falls below a tolerance ϵ:

$$\text{Acc} = \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} \mathbf{1}\Big[\text{err}(\hat{h}_\tau; \mathcal{D}_\tau^{\text{test}}) < \epsilon\Big]. \tag{11}$$

On NewtonBench, ActiveSciBench-Chem and ODEBench, err is the root mean squared logarithmic error

$$\text{RMSLE}(\hat{h}; \mathcal{D}^{\text{test}}) = \Big(\frac{1}{|\mathcal{D}^{\text{test}}|} \sum_{(u,y) \in \mathcal{D}^{\text{test}}} \big(\ell(\hat{h}(u)) - \ell(y)\big)^2\Big)^{1/2}, \tag{12}$$

which suits outputs that span several orders of magnitude. For NewtonBench and ActiveSciBench-Chem, $\ell(z) = \log(1 + |z|)$ and $\mathcal{D}^{\text{test}}$ is 1000 inputs drawn at random from the task's input ranges. For ODEBench, $\hat{h}$ is each component $\hat{f}_i$ of the recovered vector field, $\mathcal{D}^{\text{test}}$ is the states along the true trajectories from five held-out initial conditions (250 time points on $[0, 10]$: 100 on $[0, 1]$ and 150 on $(1, 10]$), the target is $\mathrm{d}x_i/\mathrm{d}t$, and $\ell(z) = \text{sign}(z) \log(1 + |z|)$ because derivatives can be negative. A system counts only if every component passes. On ForceBench, $\hat{h}_\tau$ is integrated from each held-out launch $u \in \mathcal{D}^{\text{test}}$, and err is the normalized mean squared error

$$\text{NMSE}(\hat{h}; \mathcal{D}^{\text{test}}) = \frac{\sum_{u \in \mathcal{D}^{\text{test}}} \sum_{t=1}^{T} \|\hat{\mathbf{x}}_u(t) - \mathbf{x}_u(t)\|^2}{\sum_{u \in \mathcal{D}^{\text{test}}} \sum_{t=1}^{T} \|\mathbf{x}_u(t) - \bar{\mathbf{x}}\|^2}, \tag{13}$$

the squared position error over the $T = 10$ measurement times, divided by the spread of the true positions. An NMSE of 1 is the error of always predicting the mean position. The tolerances are $\epsilon = 0.01$ (NewtonBench), 0.05 (ActiveSciBench-Chem), 0.1 (ForceBench) and 1×10^{-5} (ODEBench). ForceBench's held-out launches use more extreme particle properties than any training experiment, so passing there requires extrapolation. On the other three benchmarks the evaluation points are new but lie within the ranges the agent could query.

[4] https://github.com/scientific-discovery/LLM-ACES

[5] https://github.com/scientific-discovery/LLM-AutoSciLab

[6] https://github.com/SampsonML/DiscoverPhysics

B COREAGENT

B.1 LLM-GUIDED HYPOTHESIS SEARCH

A `Search` step advances the exploration and exploitation populations once and therefore uses two LLM requests, one per population. Each request returns two candidate expressions or proposals which are then used to define the variable-operator search space for a subsequent PySR search (Cranmer, 2023). Both the original proposal and the PySR refined expression are retained when valid (finite predictions on $\mathcal{D}_t$) and structurally novel (their constant-free skeleton is not already in the population). The two populations differ primarily in sampling behavior. The exploration population samples at temperature 1.0 and top-p 0.95 and asks for a structure unlike its parents. The exploitation population samples at temperature 0.7 and top-p 0.8 and asks for an improved version of its parents. In both cases, the LLM is conditioned on three parents sampled from the corresponding population. Evidence affects subsequent proposals through parent selection, ruled-out structures, and a compact input-relevance summary, all recomputed after each experiment. Hypotheses are ranked using Eq. 6 with $\mathcal{L}_t$ the RMSLE on $\mathcal{D}_t$ and $\tau = 0.5$, where

$$C_t(h) = \frac{k(h)\log n + 0.05|h|}{n}, \qquad n = \max(n_t, 2),$$

Here $k(h)$ is the number of fitted constants and $|h|$ the number of nodes in the parsed expression. The penalty shrinks as data accumulate. The current best is the hypothesis with the highest S_t, and ties go to the shorter expression. Parents are sampled as in LLM-SR (Shojaee et al., 2025). Every five `Search` steps, five hypotheses are drawn at random from each population and paired. Each pair swaps populations with probability

$$\alpha = \min\Big\{1, \Big(\frac{\mathcal{L}_t(h_{\text{exploit}})}{\mathcal{L}_t(h_{\text{explore}})}\Big)^{\gamma}\Big\}, \qquad \gamma = \xi\,(\beta_{\text{exploit}} - \beta_{\text{explore}}),$$

with losses clipped to $[10^{-12}, 10^{12}]$. Following Wang et al. (2026), $\beta_{\text{exploit}} = 0.8$ and $\beta_{\text{explore}} = 0.01$ are fixed inverse temperatures of the two populations, and ξ is a scale factor tuned during the run. ξ starts at 2.5. After every ten proposed pairs, it is multiplied by 1.1 if more than 40% were accepted and by 0.9 if fewer than 20% were, and is kept within $[0.05, 50]$. This holds the acceptance rate near 30%; a larger ξ makes swaps of worse-fitting exploratory hypotheses rarer.

B.2 HYPOTHESIS-GUIDED EXPERIMENT DESIGN

Competitive subset. The belief covers at most $K = 12$ hypotheses taken from both populations. We keep the best-scoring hypothesis of each structure, remove those scoring more than 0.15 below the best (S_t, Eq. 6), and remove any whose predictions nearly match those of a better one (average difference below 0.02 on the log scale g). If fewer than five remain, the next-best hypotheses are added back until there are five. All others get zero weight.

Belief. Each hypothesis predicts the outcome up to Gaussian error on the log scale,

$$p(y \mid h, u) = \mathcal{N}\big(g(y);\, g(h(u)),\, \sigma_t^2\big), \qquad \sigma_t = \max\big(\min_h \mathcal{L}_t(h),\, 0.02\big),$$

so the error level is set by the best current fit. This is the $p(y \mid h, u)$ used in F_t (Eq. 5). Starting from equal prior weights, the belief is

$$\log q_t(h) = -\tfrac{1}{2}\, n_{\text{eff}} \Big(\frac{\mathcal{L}_t(h)}{\sigma_t}\Big)^2 - \tfrac{1}{2}\, k(h) \log n_{\text{eff}} + \text{const},$$

where $k(h)$ is the number of fitted constants. The first term rewards fit and the second penalizes constants, as in BIC (Schwarz, 1978). We fix $n_{\text{eff}} = 8$ instead of using the number of observations; otherwise the belief would settle on one hypothesis after only a few experiments.

Choosing the experiment. The simulators have no noise, so if h is true, the result of experiment u is exactly $h(u)$, and Eq. 8 can be computed exactly:

$$I_t(u) = \mathbb{H}[q_t] - \sum_h q_t(h)\, \mathbb{H}\big[q_t(\cdot \mid u, h(u))\big], \qquad q_t(h' \mid u, y) \propto q_t(h')\, p(y \mid h', u).$$

Every experiment costs the same, so we pick the u with the largest $I_t(u)$. Because I_t is at most $\log K$, `Exp` is allowed only when at least five hypotheses are in the subset. The candidates $\mathcal{U}_t$ are input settings drawn at random within the allowed ranges (uniformly on a log scale), redrawn after each experiment.

B.3 Budget-Aware Search–Experiment Metareasoning

Value model. The state ϕ_t has 29 features summarizing fit, search progress, diversity, belief uncertainty, information gain and budget use, standardized within the run. For each action a, $Q_a(s_t) = m_a + \phi_t^\top w_a$ is fitted by Bayesian linear regression with prior $w_a \sim \mathcal{N}(0, 0.5^2 I)$ and noise standard deviation 0.25 on each credit. The prior means are $m_{\texttt{Search}} = 0.02$ and $m_{\texttt{Exp}} = 0$ nats, so with no data `Search` is slightly preferred. At each decision, Thompson sampling (Thompson, 1933) draws a value for each allowed action from its posterior and takes the larger. The model starts from this prior in every run; nothing is carried over between tasks, benchmarks or runs.

Credit Assignment. F_t^- and F_t^+ are the predictive distributions (Eq. 5) of two saved copies of the population. For a `Search` step, F_t^- is the population just before the step and F_t^+ the population when the next experiment is chosen. For an experiment, F_t^- is the population when it is chosen and F_t^+ the population after its observation is added and all hypotheses are refit. Both are scored on $\mathcal{T}_t$, the observations returned by the next experiment, which neither has seen. The costs are $c_S = 0.01$ nats per `Search` step and $c_E(1 + j/B^E)$ for the j-th experiment, with $c_E = 0.05$, so later experiments must gain more. For a `Search` step, κ_t is c_S times the number of `Search` steps from it to the next experiment; for an experiment, it is that experiment's cost. Credits are clipped to $[-5, 5]$; a credit that is not finite (e.g. when F_t^- has no hypotheses, as for the first `Search` step of a run) is set to -5. Credits are computed, and the posteriors updated, only when an experiment returns, so the last experiment of a run is never credited. The utility U is never computed: ν in Eq. 5 is the distribution of held-out conditions, and G_t instead measures the change in log loss on the conditions the next experiment actually visits.

Allowed actions. `Search` is allowed when at least two LLM requests remain and fewer than 20 `Search` steps have been taken since the last experiment. `Exp` is allowed when experiments remain, the competitive subset has at least five hypotheses, and at least two `Search` steps have been taken since the last experiment; this last rule is dropped when `Search` is not allowed. Until an action has two credits, it is taken whenever allowed (`Search` first). As a result, every run starts the same way: the first experiment comes after 20 `Search` steps, and the second and third after two `Search` steps each. Thompson sampling decides from then on. If neither action is allowed, which can happen only early in a run, one space-filling experiment is run at the candidate farthest from all conditions measured so far. A run terminates when convergence is sustained across three consecutive experiments and confirmed by a final space-filling audit in an under-sampled region, or when either resource budget is exhausted.

C Additional Results

C.1 Stratified Performance Analysis

NewtonBench. Figure 5 stratifies the 108 tasks into easy, medium, and hard levels (36 each). CoREAgent achieves the highest exact accuracy at every level, reaching 94.4%, 91.7%, and 77.8%, respectively. Its margin over the strongest baseline increases with difficulty, from 5.6 points on easy tasks to 16.7 points on hard tasks. Moreover, CoREAgent reaches 50% accuracy on hard tasks within five experiments, whereas LLM-AutoSciLab requires 16 and LLM-only/LLM+BED never reach this level within the budget. The metareasoner also allocates more experiments to harder tasks, averaging 12.8, 14.2, and 15.3 acquisitions across the three levels.

ODEBench. Figure 6 groups the 61 systems into 1D, 2D, and 3D dynamics. CoREAgent shows its largest advantage on multi-dimensional systems, recovering 96.4% of 2D systems compared with 78.6% for LLM-ACES and Active PySR, while reaching 67.9% after only one experiment. On 3D systems, all three methods reach 80%, but CoREAgent reaches this level after three experiments, compared with five for Active PySR and six for LLM-ACES. On 1D systems, LLM-ACES is ahead by one task (65.2% vs. 60.9%). CoREAgent also requires fewer experiments as dimensionality increases, averaging 9.7, 8.4, and 7.6 experiments for 1D, 2D, and 3D systems.

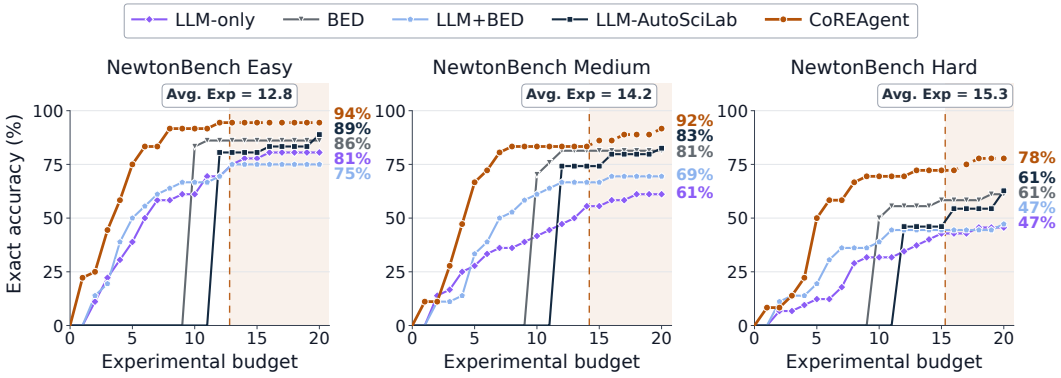


Figure 5: **NewtonBench performance by difficulty.** Dashed lines indicate the mean stopping point for CoREAgent.

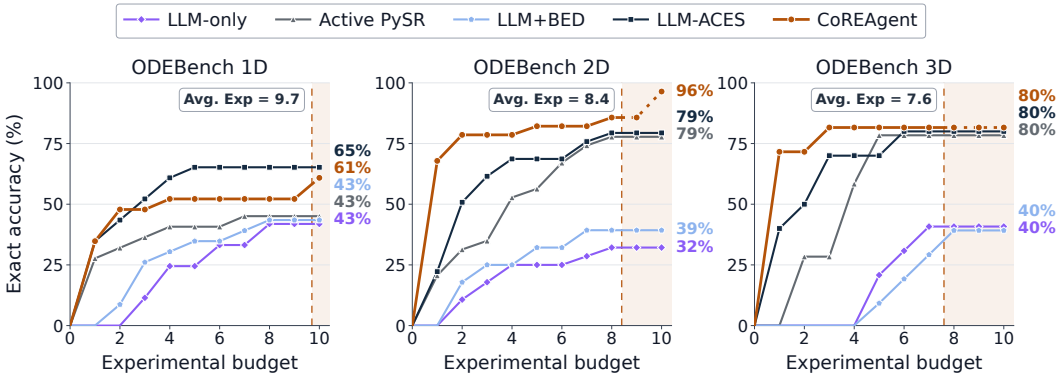


Figure 6: **ODEBench performance by state dimension.** Dashed lines indicate the mean stopping point for CoREAgent.

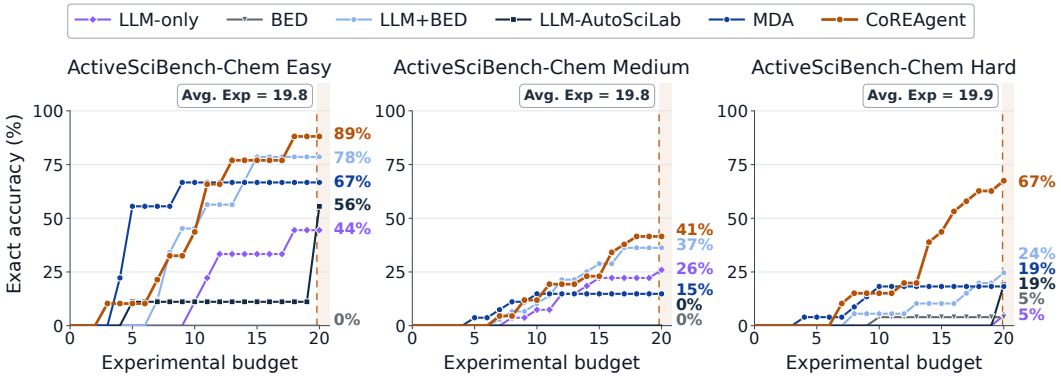


Figure 7: **ActiveSciBench-Chem performance by mechanism tier.** Dashed lines indicate the mean stopping point for CoREAgent.

ActiveSciBench-Chem. We adopt the difficulty tiers of ActiveSciBench-Chem provided by LLM-AutoSciLab (Kabra et al., 2026). Figure 7 shows the stratified results on the 57 tasks divided into easy, medium, and hard mechanism tiers. CoREAgent achieves the highest exact accuracy in all three tiers, reaching 88.9%, 40.7%, and 66.7%, respectively. The largest gain occurs on hard mechanisms, where it recovers nearly three times as many tasks as the strongest baseline (66.7% vs. 23.8%). Interestingly, the medium tier is empirically the hardest across methods, suggesting that isolating multiple interacting modifiers can be more difficult than identifying a single nonstandard mechanism. Unlike NewtonBench and ODEBench, CoREAgent uses nearly the full experimental budget across all tiers (19.8–19.9 experiments), reflecting the greater difficulty of ActiveSciBench-Chem.

C.2 Accuracy–Cost Trade-off

Figure 8 compares final exact accuracy against total discovery cost. To combine the two action types, we weight them by the common resource budgets: 400 search calls versus 20 experiments, giving an experiment-to-search cost ratio of 20:1. For each method, total cost is $C = N_S/400 + N_E/20$, where N_S and N_E are its mean numbers of search calls and experiments across all tasks in the benchmark. We plot C divided by its maximum value of 2 (both budgets exhausted), so normalized cost lies in $[0, 1]$ and cost ratios are preserved. This reporting cost is used only to compare total resource consumption and is distinct from the utility penalties c_S and c_E used by the metareasoner. Exact accuracy is plotted as a fraction in $[0, 1]$ and reported as a percentage in the text. On NewtonBench, CoREAgent is the only Pareto-optimal method, attaining the highest accuracy at the lowest cost. On ActiveSciBench-Chem, MDA and CoREAgent are both Pareto-optimal. MDA operates at lower cost, using minimal search and a benchmark-specific configuration, but CoREAgent raises exact accuracy from 24.6% ($14/57$ laws) to 57.9% ($33/57$), a $2.35\times$ improvement, for 37% additional total cost. Since both methods use nearly the full experimental budget, this gap reflects more effective use of search rather than additional data acquisition. The strongest remaining baseline, LLM+BED, recovers $22/57$ laws at $1.44\times$ the cost of CoREAgent.

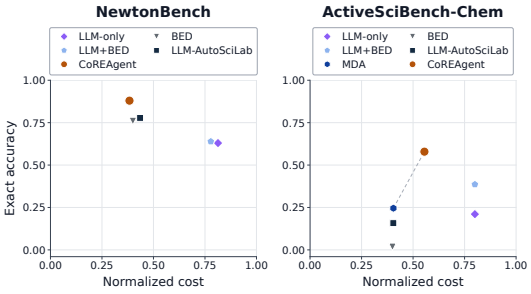


Figure 8: **Accuracy–cost trade-off.** Exact accuracy (as a fraction in $[0, 1]$) versus normalized total cost. A method is Pareto-optimal if no other method has both equal-or-higher accuracy and equal-or-lower cost, with at least one strict improvement.

C.3 Generalization Across LLMs

To test the generalizability of CoREAgent across LLM backbones, we evaluate three model families: GPT-4o-mini, Qwen3-32B, and Claude Haiku 4.5, keeping all other components fixed. We use 25 randomly stratified tasks, 15 from NewtonBench and 10 from ActiveSciBench-Chem. As shown in Table 4, every backbone recovers at least two-thirds of the NewtonBench laws ($10/15$). On ActiveSciBench-Chem, Qwen3-32B achieves the highest accuracy (60.0%), ahead of GPT-4o-mini (50.0%), showing that the performance is not tied to a single proprietary model. Experiment usage remains similar across backbones, ranging from 14.1–15.4 on NewtonBench and 19.8–20.0 on ActiveSciBench-Chem. The metareasoner directly makes the allocation decisions, while the backbone can influence them indirectly through the hypotheses it generates. Despite this, experiment usage remains similar across the three backbones. Backbone choice has a larger effect on ActiveSciBench-Chem, where Claude Haiku 4.5 recovers 30.0% of mechanisms compared with 50.0–60.0% for the other two models, suggesting that harder mechanisms are more sensitive to proposal quality while the overall allocation behavior remains stable.

Table 4: **CoREAgent with different LLM backbones.** We report the results using Exact accuracy (%) and mean number of experiments. Best value per column in **bold**.

LLM Backbone	NewtonBench		ActiveSciBench-Chem	
	Acc. ↑	Exp. ↓	Acc. ↑	Exp. ↓
GPT-4o-mini	**80.0**	14.9	50.0	20.0
Qwen3-32B	66.7	15.4	**60.0**	20.0
Claude (Haiku 4.5)	**80.0**	**14.1**	30.0	**19.8**