PRINCIPIA: TEST-TIME HYPOTHESES TO PRINCIPLE CONSTRUCTION FOR SCIENTIFIC DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Scientific discovery requires identifying the scientific principles underlying observed phenomena to guide further investigation. As these empirical principles are often unknown in complex discovery processes, they are constructed via a hypotheses-to-principle pipeline to obtain recurring causalities from progressively accumulated observations during test-time discovery. However, constructing this knowledge requires substantial domain expertise, time, and effort, making this translation stage a bottleneck for scientific discovery. To address this limitation, we propose **PRINCIPIA**, an LLM-based framework that constructs and continually refines scientific principles at test-time. Drawing on how human researchers develop principles, PRINCIPIA forms comparative causal hypotheses from related inputs and their observed outcomes, then synthesizes recurring relationships across these hypotheses into inductive principles. By applying the constructed hypotheses and inductive principles to the task, it obtains new observations, revises the used hypotheses based on these new observations, and reconstructs principles from the updated hypotheses, allowing scientific knowledge to continuously evolve throughout test-time discovery. Across benchmarks spanning molecular optimization, protein engineering, gene perturbation, and molecular property prediction, PRINCIPIA outperforms baselines relying on internal knowledge, external sources, or experience summaries, while generating principles that are plausible and consistent to scientific literature, demonstrating its effectiveness in constructing and refining task-specific hypotheses and principles at test-time for scientific discovery.

1 INTRODUCTION

Scientific discovery has long driven human progress, enabling advances across chemistry, biology, and medicine. Central to this process is identifying scientific principles that govern the systems of interest. In complex scientific discovery processes such as drug discovery, the scientific principles are usually unknown due to the complexity of the underlying systems. Therefore, empirical principles are usually constructed from progressively accumulated experimental observations during test-time discovery to explain causality between input characteristics and experimental outcomes (Scheiner & Willig, 2008; Hone et al., 2015; Pearlstein et al., 2021). Specifically, hypotheses that explain a specific scope of individual observations are formed and then generalized into principles that cover a broader scope of observations, being formulated as a hypotheses-to-principle construction pipeline (Lamma et al., 2000; Kakas & Michael, 2020; Engelschalt et al., 2023).

However, constructing such principles by human researchers requires substantial domain expertise, time, and effort, creating a bottleneck in scientific discovery. As illustrated in Fig. 1, human researchers also follow the hypotheses-to-principle construction pipeline, while leveraging an extensive body of prior knowledge. They interpret the experimental observations, hypothesize the plausible causal relationships, generalize them by analyzing the recurring patterns, and revise hypotheses when new observations contradict prior explanations. This continued reliance on expert interpretation limits the pace of scientific discovery by slowing the translation of accumulated observations into scientific principles. Recent advances in large language models (LLMs) (Anthropic, 2026; OpenAI, 2026a;b; Xu et al., 2026) have motivated growing efforts to automate scientific research (AI4Science & Quantum, 2023), including idea generation (Wang et al., 2024; Li et al., 2025; Baek et al., 2025), experimental design (Gottweis et al., 2026; Schmidgall et al., 2026), and candidate optimization (Wang et al., 2025a;b; Roohani et al., 2025; Ran et al., 2025). Despite these advances, constructing and

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.

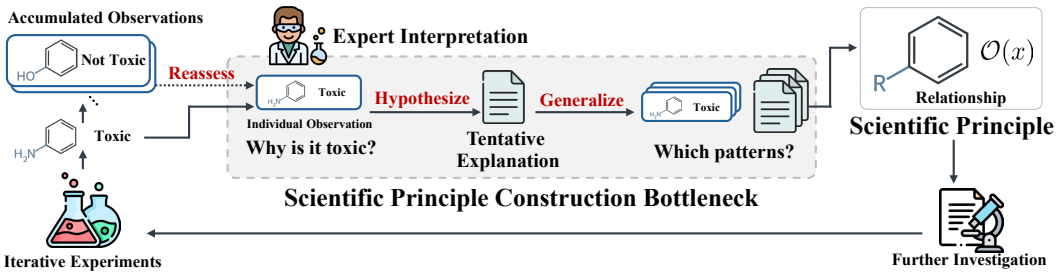


Figure 1: **Scientific principle construction as a bottleneck in scientific discovery.** As experimental observations accumulate, researchers formulate tentative explanations of individual outcomes, generalize recurring relationships across observations into scientific principles, and reassess these explanations in light of new observations. The resulting principles guide further investigation, forming an iterative discovery cycle. However, this process requires substantial domain expertise, time, labor, and financial resources, limiting the pace of scientific discovery.

continually refining principles for scientific discovery from progressively accumulated experimental observations remains underexplored.

In this work, we explore the test-time construction of empirical principles that explain how input characteristics contribute to observed outcomes using LLMs. However, operationalizing this process presents two challenges. First, an individual input and its observed outcome provide limited guidance for an LLM to distinguish among possible explanations. Multiple input characteristics can account for the same outcome, making it difficult to identify which factors to emphasize when generating a hypothesis. Hypothesis generation therefore requires evidence that helps narrow the range of possible explanations. Second, the observations accumulated during discovery cover only a small fraction of the vast space of scientific inputs. LLM-generated hypotheses based on this limited coverage can provide incomplete or incorrect explanations for other inputs, and these limitations can carry over into the principles constructed from them. As new observations accumulate, they can expose these limitations, making it necessary to reassess and revise earlier hypotheses.

To address these challenges, we propose **PRINCIPIA**, an LLM-based hypotheses-to-principle construction framework that inductively derives principles from progressively accumulated experimental observations during test-time discovery. Drawing on how human researchers develop scientific knowledge, PRINCIPIA consists of three key components: (i) comparative causal hypothesis generation, (ii) inductive principle construction, and (iii) hypothesis revision via new observations. Specifically, for hypothesis generation, PRINCIPIA compares related inputs and their experimental outcomes rather than a single observation to formulate causal hypotheses, narrowing the factors considered to their shared and differing characteristics. For principle construction, it groups related inputs and inductively combines recurring relationships across their hypotheses into principles. These hypotheses and principles guide subsequent prediction or optimization through retrieval, getting new observations. Finally, new observations are used to revise the used hypotheses and reconstruct the resulting principles, allowing the knowledge to evolve during test-time discovery.

We evaluate PRINCIPIA on extensive benchmarks of scientific discovery tasks, spanning molecular, protein, and gene optimization, as well as molecular property prediction. Our results demonstrate that PRINCIPIA outperforms baselines that rely on internal knowledge, experience summaries, or external knowledge, demonstrating the effectiveness of test-time construction of hypotheses and principles. Further analyses show that PRINCIPIA achieves comparable optimization performance with a 55% reduction in experimental budget compared with an experience-summary baseline, demonstrating its cost-effectiveness in scientific optimization. Qualitative analyses show that the constructed hypotheses capture relationships between inputs and experimental outcomes, while the induced principles are consistent with scientific literature. These results show that constructing empirical principles at test time can improve both the effectiveness and efficiency of scientific discovery.

2 RELATED WORKS

Automated Scientific Discovery and Optimization Advances in machine learning have emerged as powerful tools for automating scientific discovery, ranging from systems that assist individual

stages, such as hypothesis and idea generation (Wang et al., 2024; Vasu et al., 2025; Li et al., 2025; Baek et al., 2025), to end-to-end AI scientists (Schmidgall et al., 2025; 2026; Lu et al., 2026; Gottweis et al., 2026) that formulate research proposals, conduct experiments, and generate scientific papers with minimal human supervision. While these works seek to automate the broader research workflow, another line of research focuses on discovering novel scientific candidates with minimal prior context. For example, Bagal et al. (2021); Xu et al. (2022); Lee et al. (2023); Watson et al. (2023) develop deep generative models that explore latent spaces to identify promising candidates by learning representations of the underlying scientific domains. Meanwhile, with the emergence of Large Language Models (LLMs), agentic approaches for candidate generation by LLMs have been developed (M. Bran et al., 2024; Roohani et al., 2025). Building on this direction, recent approaches leverage the reasoning capabilities of LLMs, using them as surrogate operators in Evolutionary Algorithms (Holland, 1962) to guide exploration of vast search spaces through iterative generation and evolutionary search (Romera-Paredes et al., 2024; Wang et al., 2025a;b; Novikov et al., 2025; Yuksekgonul et al., 2026; Lange et al., 2026; Zhang et al., 2026a) across diverse scientific research domains. Despite their progress, constructing reusable scientific knowledge from experimental observations during discovery remains underexplored. Our work addresses this gap by dynamically constructing reusable causal principles at test time, enabling more informed discovery.

Knowledge-Augmented Scientific Discovery Building on the success of Retrieval-Augmented Generation (RAG) (Lewis et al., 2020), a growing body of work has incorporated external knowledge into scientific discovery systems. These approaches include leveraging existing literature to generate open-ended hypotheses and research proposals through multi-agent systems (Wang et al., 2024; Li et al., 2025; Baek et al., 2025; Huang et al., 2025; Liu et al., 2025), incorporating external knowledge bases into the discovery loop to inform experimental decisions (Boiko et al., 2023; M. Bran et al., 2024; Roohani et al., 2025), and employing deep-search agents to retrieve promising research directions (Liu et al., 2025). Recent work has also explored using experience during discovery to guide subsequent exploration. For instance, Pu et al. (2025) balance exploration and exploitation over pre-configured high-quality scientific knowledge to guide subsequent optimization of both candidates and principles, while Ran et al. (2025) summarize characteristics of previously discovered candidates into a single document. Another line of works leverages predictive strategies for machine learning research (Jiang et al., 2026; Luo et al., 2026) or constructs offline memories from heterogeneous tasks for future reuse (Zhang et al., 2026b; Yao et al., 2026). Beyond retrieving literature or reusing summarized experience, our work focuses on causal principle induction from observations accumulated during test-time discovery. Rather than using these observations solely to guide subsequent actions, we develop explicit hypotheses about how input characteristics may contribute to observed outcomes and generalize recurring relationships into principles.

3 PRINCIPIA: TEST-TIME HYPOTHESES TO PRINCIPLE CONSTRUCTION

We present PRINCIPIA, an LLM-based framework that constructs and refines empirical principles at test time by forming and generalizing hypotheses from progressively accumulated experimental observations (Fig. 2). We first introduce comparative causal hypotheses formed from related inputs and their observed outcomes, inductive principle construction that generalizes recurring patterns across these hypotheses, and hypotheses revision that applies the constructed hypotheses and principles to get new observations and updates the previously formed hypotheses via new observations.

Task Settings We consider two central tasks in scientific discovery: optimization and prediction. Let $x \in \mathcal{X}$ denote a scientific input, such as a molecule, and let $\mathcal{O}$ denote an experimental oracle that returns an observed property or characteristic $\mathcal{O}(x)$. We refer to this outcome as a *scientific observation*, which could be a categorical label or a numerical measurement. In the optimization task, the goal is to find an optimal instance x^* that best achieves a specified scientific objective such as maximizing a desired property, through successive modifications. Given an input x, the action is to propose a modified input x', which is evaluated by the experimental oracle to obtain the observation $\mathcal{O}(x')$. In the prediction task, the goal is to predict a property or characteristic of a given input x. The action is to produce a prediction $\hat{\mathcal{O}}(x)$, which is evaluated against the observation $\mathcal{O}(x)$ without access to that observation during prediction. In both settings, the observations provide evidence for constructing and refining the knowledge that guides subsequent modifications or predictions.

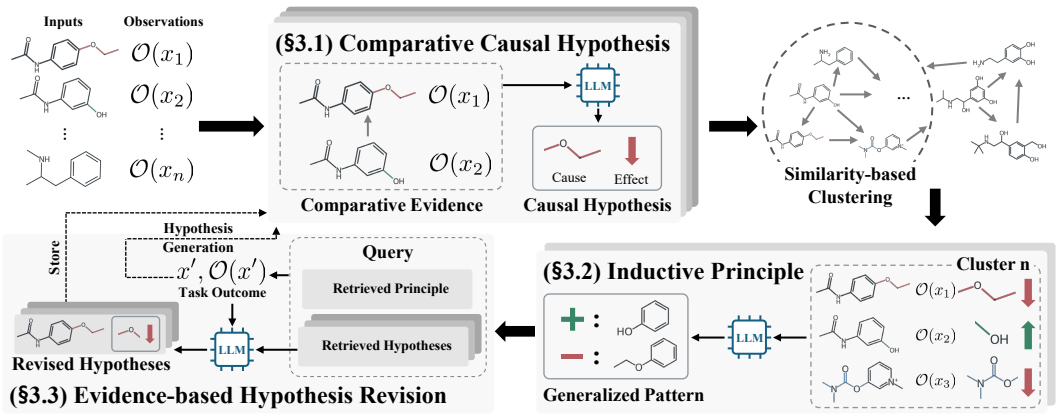


Figure 2: **Overview of PRINCIPIA.** Given scientific inputs and their observed outcomes, PRINCIPIA forms comparative causal hypotheses that explain how input characteristics may contribute to the outcomes. It groups related inputs through similarity-based clustering and inductively generalizes their associated hypotheses into scientific principles. Retrieved hypotheses and principles guide subsequent prediction or candidate generation. New observations are then used to revise the hypotheses involved, and the updated hypotheses support principle reconstruction, forming a continuous cycle of scientific knowledge construction and application.

3.1 COMPARATIVE CAUSAL HYPOTHESIS

We now turn to the first step of our scientific knowledge construction, which is the formation of causal hypotheses. The goal of hypothesis formation is to propose explicit explanations of the factors that have contributed to an observed experimental outcome. However, an individual input and its observed outcome alone generally provide limited evidence for distinguishing the contributions of different input characteristics. To provide a more reliable basis for hypothesis formation, we construct comparative evidence by combining related inputs with their corresponding observed outcomes. From this evidence, we generate a *comparative causal hypothesis*, defined as a tentative explanation of how similarities and differences between two inputs account for their observed outcomes.

Comparative Evidence To narrow the scope of comparison, we first form comparative evidence which consists of two similar inputs and their experimental outcomes. Specifically, we compose reference-to-target comparisons $x_i \rightarrow x_j$ from the available experimental data, $\mathcal{D} = \{(x_i, \mathcal{O}(x_i))\}_{i=1}^{N}$, by selecting related inputs in the sense that they share similar input characteristics. In the optimization task, we compare a generated candidate x_j with the input x_i from which it was derived, using their observed outcomes to examine the effects of the modification. In the prediction task, such modification histories are not generally available, so we compute the pairwise distances between each target input x_j and nearby reference inputs x_i from $\mathcal{D}$. To compute the pairwise distances, we leverage domain-appropriate distance measures, such as fingerprint-based distance for molecules, and select references with the smallest distances, regardless of whether their observed labels match the target's label. In both settings, each pair provides comparative evidence consisting of both inputs and their observations: $e_{i\rightarrow j} = \big(x_i, \mathcal{O}(x_i), x_j, \mathcal{O}(x_j)\big)$.

Hypothesis Generation Then, an LLM generates textual hypotheses $H_{i\rightarrow j}$ from the comparative evidence and a task description: $H_{i\rightarrow j} = \text{LLM}_{\text{hypothesis}}\big(e_{i\rightarrow j}; q\big)$, where q is the task description. Based on this comparative evidence, an LLM identifies the similarities and differences between the inputs and their observations and links an input characteristic to their observations, proposing a possible explanation of the causal relationship grounded in the comparison. Consequently, when outcomes differ, the hypotheses focus on differences between the inputs that can explain the observed outcomes. Otherwise, they focus on describing shared characteristics that can maintain the observed outcomes. We then store each hypothesis together with its comparative evidence, $(e_{i\rightarrow j}, H_{i\rightarrow j})$, in a graph-structured hypothesis memory $\mathcal{M}_h$, where nodes represent inputs and each directed edge stores the comparative evidence and its associated causal hypothesis for a reference-to-target comparison, building scientific knowledge specific to the inputs and observations involved.

3.2 Inductive Principle

Even though each comparative causal hypothesis provides a tentative explanation grounded in a particular pair of inputs, a single hypothesis could depend on characteristics specific to those inputs, failing to model the general underlying scientific principles. Inspired by the human scientific process where the scientist generalizes from specific observations to broader principles, we construct *inductive principles* that summarize recurring relationships between input characteristics and observed outcomes across multiple comparisons.

Similarity-based Clustering We first group the inputs in $\mathcal{D}$ to examine recurring patterns among inputs with similar characteristics by applying k-medoids clustering with the domain-specific pairwise distances. Let G_k denote the set of input indices in the k-th cluster and $\mathcal{A}_h$ denote the directed edge set of the hypothesis graph $\mathcal{M}_h$. For each cluster, we collect the comparative evidence and causal hypotheses on all edges incident to its members: $\mathcal{E}_k = \{(e_{i\to j}, H_{i\to j}) \mid (i,j) \in \mathcal{A}_h, i \in G_k \vee j \in G_k\}$. This grouping brings together inputs with similar characteristics and the hypotheses associated with their comparisons, providing a basis for identifying recurring relationships across observations.

Principle Induction For each cluster, an LLM inductively infers a compact principle based on its associated evidence and hypotheses: $P_k = \mathrm{LLM}_{\text{principle}}\big(\mathcal{E}_k; q\big)$. The model identifies recurring relationships across the associated causal hypotheses and generalizes them into a causal principle. By integrating these across multiple comparisons, this process extends beyond individual observations to derive a causal principle that captures more general relationships. We store each principle with its associated cluster in a principle memory $\mathcal{M}_p = \{(G_k, P_k)\}_{k=1}^{K}$, organizing the induced principles into a more general level of scientific knowledge.

3.3 Hypothesis Revision via New Observations

Thus far, we have described how to generate comparative causal hypotheses and inductive principles from experimental observations: the inductive principles are constructed from comparative causal hypotheses, making the reliability of these principles dependent on the hypotheses underlying them. However, experimental observations cover only a small fraction of the space of possible inputs, and hypotheses formed from this limited evidence can overlook factors relevant to other inputs. These limitations can also affect the principles constructed from them, leading to potentially inaccurate or incomplete conclusions. Therefore, we revise hypotheses to enhance their reliability and improve the quality of the principles constructed from them. Specifically, inspired by how scientists validate existing hypotheses through further investigation, we use the generated hypotheses and principles to guide further task-specific predictions or candidate generation, obtaining additional experimental observations. We then revise the hypotheses using new observations, either by comparing predictions with observed labels or by examining the evaluated outcomes of generated candidates.

Hypothesis Application and Revision To obtain new observations for hypothesis revision, we first retrieve relevant hypotheses and principles and apply them to new inputs. Given a query input x, we first construct hypothesis and principle candidate pools drawn from $\mathcal{M}_h$ and $\mathcal{M}_p$, and then leverage an LLM-based retrieval, where an LLM that can handle long-context inputs independently selects k_h hypotheses $\mathcal{R}_h(x)$ and k_p principles $\mathcal{R}_p(x)$ from candidate pools, respectively, based on their relevance to the query and task description. The selected knowledge guides the task-specific action such as optimization or prediction, after which the corresponding observations are obtained by these actions. For more details on the retrieval and application of knowledge, please refer to Appendix A.

We then revise the hypotheses retrieved and used during the current round. For each selected hypothesis $H_{i\to j}$, we construct new observation $\mathcal{B}_{i\to j}$ by sampling new observations from that round together with their associated inputs and outcomes. Specifically, in the optimization task, $\mathcal{B}_{i\to j}$ contains input pairs (x, x') by the modification from x to x' and their observed outcomes $(\mathcal{O}(x), \mathcal{O}(x'))$. In the prediction task, it contains query inputs x, their observed labels $\mathcal{O}(x)$, and predictions $\hat{\mathcal{O}}(x)$ made without access to those labels. An LLM then revises each selected hypothesis using new observations, $\mathcal{B}_{i\to j}$, and the task description: $H'_{i\to j} = \mathrm{LLM}_{\text{refine}}\big(e_{i\to j}, H_{i\to j}, \mathcal{B}_{i\to j}; q\big)$. By grounding revision in new observations obtained from applying the knowledge to new inputs, the framework provides an empirical basis for assessing and refining the hypotheses beyond the comparisons from which they were originally formed.

Knowledge Update After revision from additional observations, we replace $H_{i\to j}$ with its revised version $H'_{i\to j}$ in $\mathcal{M}_h$, while retaining the original comparative evidence $e_{i\to j}$. Then, the new observations and inputs are also used for generating their hypotheses following the procedure in Section 3.1. This extends the hypothesis memory with new observations while revision updates existing hypotheses. We then reconstruct principles from the updated hypotheses and associated observations following the procedure in Section 3.2. The resulting knowledge memories guide subsequent investigation, connecting observation, hypothesis generation, knowledge application, and revision via new observations in a continuous loop.

4 Experimental Results

We evaluate PRINCIPIA on scientific discovery tasks including scientific optimization tasks spanning molecules, proteins, and genes, as well as molecular property prediction. We provide implementation details in Appendix A and the algorithmic workflow for optimization and prediction in Appendix D.

4.1 Scientific Optimization

Experimental Setting We evaluate PRINCIPIA on scientific optimization tasks under a restricted oracle budget spanning molecules, proteins, and genes from the Scientific Discovery Evaluation benchmark (Song et al., 2025), involving designing molecules with desired properties (Gao et al., 2022), optimizing protein sequences for target functions (Wang et al., 2025b), and identifying gene perturbations that produce desired cellular responses (Schmidt et al., 2022; Scharenberg et al., 2023). We report AUC@10 to measure optimization progress over oracle evaluations and the average oracle scores of the top 100 molecules or top 50 protein sequences. For gene perturbation, we additionally report hit ratio following Roohani et al. (2025). To ensure fair comparisons across variants, we normalize the fitness values of common variants for the Syn-3bfo, Scharenberg22, and IFNG tasks to the range $[0, 1]$ when computing AUC. We use GPT-5-mini (Singh et al., 2025) as the backbone. We provide task-specific objectives, search budgets, and evaluation protocols in Appendix B.1 and B.2.

Baselines We compare against MolLEO (Wang et al., 2025a), an optimizer without guidance, ExLLM (Ran et al., 2025) which guides optimization by an evolving textual summary from experience, WebSearch which provides guidance retrieved from the web, DeepEvolve (Liu et al., 2025) which uses web search guidance based on the optimization experience, and PiFlow (Pu et al., 2025) which guides optimization without observation-guided causal hypotheses. For molecular optimization, we additionally include REINVENT (Olivecrona et al., 2017), a reinforcement learning method for molecular generation, Augmented Memory (Guo & Schwaller, 2023) which enhances reinforcement learning through experience replay, and Graph GA (Jensen, 2019) which applies genetic operations to molecular graphs. We provide implementation details and configurations in Appendix C.1.

Results As shown in Tables 1 and 2, PRINCIPIA consistently outperforms baseline without guidance (MolLEO), demonstrating that constructing task-specific knowledge is beneficial for guiding optimization. PRINCIPIA also outperforms baselines that use experience summaries (ExLLM), showing that constructing structured knowledge including causal hypotheses and inductive principles is more effective than a textual summary of past observations. Notably, PRINCIPIA also outperforms baselines that use external knowledge (WebSearch, DeepEvolve), indicating that constructing task-specific knowledge at test-time provides core information to achieve the optimization objective, while external knowledge is less relevant to the specific task. On the other hand, the baseline without explicit modeling of causal relationships (PiFlow) often underperforms the baseline without guidance (MolLEO), suggesting that constructing causal hypotheses and generalizing them into principles is crucial for guiding optimization. These results demonstrate the effectiveness of constructing task-specific knowledge from accumulated observations for guiding optimization under limited evaluation budgets.

4.2 Molecular Property Prediction

Experimental Setting We evaluate PRINCIPIA on molecular property prediction with eight binary classification tasks from TDC (Huang et al., 2021): four toxicity-related tasks (DILI, hERG, Ames, and Skin Reaction) and four CYP inhibition tasks (CYP1A2, CYP2C9, CYP2D6, and CYP3A4). We

Table 1: Comparison of PRINCIPIA with ML-based and LLM-based optimization baselines on molecular optimization tasks. We report averaged values over three different runs for each metric. * denotes methods reimplemented for our evaluation setting. Avg. denotes the mean of each metric across all tasks. The best results are highlighted in **bold**.

	JNK3		GSK3β		Median1		Median2		Valsartan Smarts		Avg.	
Methods	Avg@100	AUC@10	Avg@100	AUC@10	Avg@100	AUC@10	Avg@100	AUC@10	Avg@100	AUC@10	Avg@100	AUC@10
REINVENT	0.164	0.160	0.438	0.368	0.202	0.187	0.164	0.167	0.000	0.000	0.194	0.176
Memory-Aug	0.282	0.211	0.535	0.414	0.213	0.189	0.178	0.167	0.000	0.000	0.242	0.196
Graph GA	0.316	0.290	0.352	0.315	0.175	0.177	0.162	0.167	0.000	0.000	0.201	0.190
MolLEO*	0.266	0.273	0.400	0.341	0.293	0.280	0.197	0.189	0.018	0.028	0.235	0.222
ExLLM*	0.245	0.227	0.438	0.363	0.293	0.281	0.198	0.190	0.096	**0.188**	0.254	0.250
PiFlow*	0.050	0.057	0.223	0.278	0.238	0.255	0.186	0.196	0.000	0.000	0.139	0.157
WebSearch*	0.192	0.199	0.505	0.421	0.290	0.277	0.197	0.189	0.040	0.066	0.245	0.230
DeepEvolve*	0.248	0.234	0.468	0.382	0.294	0.277	0.184	0.181	0.047	0.084	0.248	0.232
PRINCIPIA (Ours)	**0.448**	**0.350**	**0.617**	**0.452**	**0.298**	**0.284**	**0.220**	**0.203**	**0.117**	0.106	**0.340**	**0.279**

Table 2: Comparison of PRINCIPIA with LLM-based optimization baselines on protein sequence engineering and gene perturbation tasks. We report averaged values over three different runs for each metric. * denotes methods reimplemented for our evaluation setting. Avg. denotes the mean AUC@10 across all tasks. The best results are highlighted in **bold**.

	Protein Sequence Engineering						Gene Perturbation				
	Syn-3bfo		AAV		GB1		Scharenberg22		IFNG		Avg.
Methods	Avg@50	AUC@10	Avg@50	AUC@10	Avg@50	AUC@10	Hit Ratio	AUC@10	Hit Ratio	AUC@10	AUC@10
MolLEO*	-0.374	0.737	0.664	0.524	0.201	0.245	0.270	0.606	0.107	0.717	0.566
ExLLM*	-0.286	0.738	0.660	0.522	0.236	0.277	0.277	0.605	0.109	0.735	0.575
PiFlow*	-32.149	0.079	0.049	0.072	0.111	0.188	0.289	0.557	0.111	0.682	0.316
WebSearch*	-0.367	0.738	0.659	**0.527**	0.197	0.233	0.226	0.533	0.110	0.702	0.547
DeepEvolve*	-0.378	0.737	0.664	0.524	0.196	0.275	0.270	0.539	0.108	0.560	0.527
PRINCIPIA (Ours)	**0.084**	**0.741**	**0.681**	**0.527**	**0.398**	**0.383**	**0.308**	**0.630**	**0.114**	**0.742**	**0.605**

use a scaffold split to construct a held-out test set containing molecular scaffolds absent from the training set. To simulate the test-time scenario for prediction tasks, we equally split the training dataset and construct knowledge by progressively observing the labels of each subset. After constructing knowledge from the training dataset, we evaluate the resulting knowledge on the held-out test set, whose labels are excluded from knowledge construction and revision. We use GPT-5.4-mini as the backbone. We report accuracy and macro-F1, averaged over three random seeds. We provide details on data preparation, round construction, and evaluation protocols in Appendix B.1 and B.3.

Baselines We compare against Direct prediction which predicts without additional context, and Few-shot prediction which conditions predictions on k examples from the whole training dataset composed of input molecules and their labels. We further consider Textual summary which summarizes sequentially observed training examples into texts, Internal principles which provides principles generated from the LLM's internal knowledge, and External principles which provides principles retrieved from web. We further provide implementation details and configurations in Appendix C.2.

Results As shown in Table 3, PRINCIPIA consistently outperforms all baselines across eight molecular property prediction tasks, demonstrating the effectiveness of constructing hypotheses and principles for molecular property prediction. In particular, its improvements over few-shot prediction suggest that organizing observations into explicit causal hypotheses and principles provides more effective guidance than supplying labeled examples alone. Since the knowledge is constructed from training observations and evaluated on held-out molecules with different structures, these results further suggest that the constructed knowledge captures relationships that extend beyond the observed examples and enhances the prediction accuracies for unseen molecules.

4.3 ANALYSES

Effect of Comparative Causal Hypotheses We analyze the effect of comparative causal hypotheses on scientific optimization. Table 4 shows that providing these hypotheses improves performance over the optimizer without hypothesis guidance, indicating that explicit hypotheses about the relationships between input characteristics and observed outcomes provide useful guidance for subsequent

Table 3: Comparison of PRINCIPIA with direct prediction, few-shot prediction, and knowledge-augmented baselines on molecular property prediction tasks. We report averaged values over three different runs for each metric. The best results are highlighted in **bold**.

	DILI		hERG		Ames		Skin		CYP1A2		CYP2C9		CYP2D6		CYP3A4		Avg.	
Methods	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Direct	52.43	44.11	51.33	37.27	58.33	57.51	45.12	43.54	63.33	60.18	67.33	64.73	68.33	66.64	60.00	54.26	58.28	53.53
Few-shot (k=1)	59.72	54.68	58.67	53.16	59.33	59.02	52.03	51.30	66.33	64.98	68.00	66.95	69.33	68.93	59.00	57.21	61.55	59.53
Few-shot (k=3)	62.15	57.95	62.00	59.76	62.33	61.03	52.85	51.49	64.67	63.06	70.33	69.75	71.33	71.31	61.00	59.63	63.33	61.75
Few-shot (k=5)	62.85	57.82	64.67	62.83	62.00	60.53	50.81	49.62	67.67	66.00	66.67	66.18	71.67	71.53	62.00	60.42	63.54	61.87
Few-shot (k=7)	62.15	57.74	64.00	62.60	62.33	60.94	50.81	48.40	66.33	64.65	69.67	69.42	71.67	71.65	64.67	62.97	63.95	62.30
Internal principles	55.21	49.74	53.67	44.26	56.67	53.52	51.63	49.45	68.33	67.34	66.33	62.67	69.67	68.90	64.33	61.91	60.73	57.22
External principles	54.51	48.13	61.00	58.52	55.67	52.18	52.44	50.77	67.67	66.17	62.00	60.71	67.33	66.91	60.67	54.99	60.16	57.30
Textual summary	61.46	56.95	64.33	62.70	57.00	55.26	51.63	50.81	67.67	64.86	70.00	67.10	70.67	70.12	63.33	60.40	63.26	61.03
PRINCIPIA (Ours)	**70.49**	**67.48**	**70.33**	**69.20**	**67.67**	**67.30**	**58.54**	**58.44**	**71.33**	**70.81**	**76.67**	**76.72**	**73.00**	**72.76**	**70.00**	**69.56**	**69.75**	**69.03**

Table 4: Analysis of the effect of causal hypotheses on protein sequence engineering tasks, including GB1 and Syn-3bfo.

Variant	AUC@10 ↑	Δ ↑
GB1		
None	0.245	—
w/ Hypotheses	**0.337**	**+0.092**
Syn-3bfo		
None	−0.019	—
w/ Hypotheses	**0.116**	**+0.135**

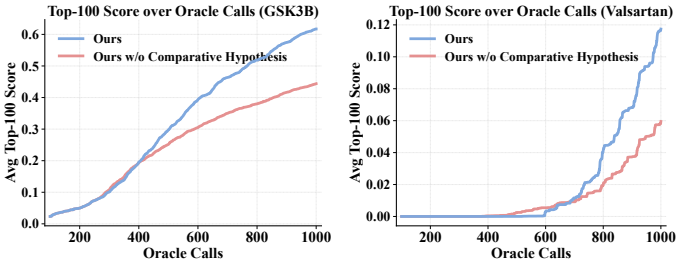


Figure 3: Analysis on the effect of comparative hypotheses generation compared to predictive hypotheses generation.

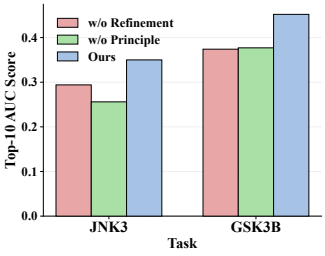


Figure 4: Analysis on the effect of Principle Induction and Evidence-based Refinement.

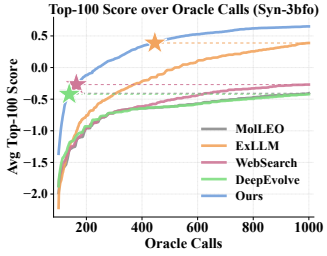


Figure 5: Analysis on sample efficiency of PRINCIPIA on Syn-3bfo optimization task.

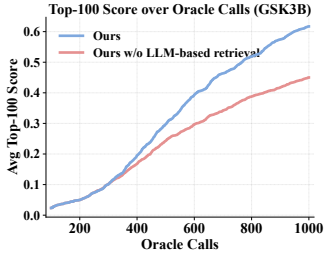


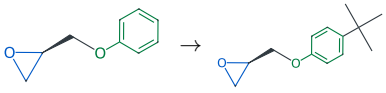
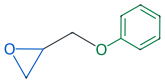
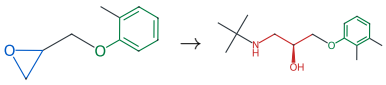
Figure 6: Analysis of the efficacy of hypothesis and principle retrieval using LLMs.

optimization. We also compare comparative hypothesis generation with a non-comparative variant that generates each hypothesis from a single input and its observed outcome (w/o comparative). As shown in Fig. 3, comparative hypothesis generation improves optimization performance, supporting our rationale that comparative evidence helps narrow the factors considered during hypothesis generation and yields more effective guidance.

Effect of Principle Construction and Hypothesis Revision To see the effect of principle construction and hypothesis revision, we ablate each component separately from PRINCIPIA. Fig. 4 shows that removing principle construction reduces performance, suggesting that generalizing individual hypotheses into principles provides critical information for subsequent discovery. Removing hypothesis revision also lowers performance, highlighting the effectiveness of reassessing existing hypotheses and reconstructing principles from the updated explanations. These results demonstrate the importance of both constructing principles from individual hypotheses and revising hypotheses based on new observations to provide up-to-date guidance for test-time scientific discovery.

Evaluation Budget Efficiency Scientific experiments can be costly, making it important to reach desired outcomes with fewer evaluations. We therefore examine whether the constructed causal principles help reduce this evaluation burden under an extended oracle budget. As shown in Fig. 5, PRINCIPIA achieves the best final performance among the compared methods. Surprisingly, PRINCIPIA matches the final performance of ExLLM, WebSearch, MolLEO, and DeepEvolve with

Table 5: An example of comparative hypotheses and an induced principle. Matching colors link epoxide rings (blue), phenoxy groups (green), and aminoalcohol side chains (red) to their descriptions. Arrows indicate reference-to-target comparisons, and labels are dataset observations.

(A) Comparative evidence	(B) Causal hypotheses	(C) Inductive principle
 **Mutagenic (+)** → **Mutagenic (+)**	The shared **aryl ether** **epoxide** motif, … is the most plausible mutagenicity driver because it is retained in both compounds.	**Positive pattern**  **Epoxide/oxirane-bearing aryl ether** motifs recur as a likely mutagenicity driver, especially Ar-O-CH2-CH …
 **Mutagenic (+)** → **Non-mutagenic (−)**	mutagenicity is driven by its **epoxide/oxirane-like ring** … has **open-chain alcohol/amine side chain**, which is far less prone to covalent DNA damage …	**Negative pattern** More polar **aminoalcohol** side chains … are often tolerated when they do not contain a strained electrophile.

approximately 55%, 84%, 86%, and 86% fewer oracle evaluations, respectively. These results suggest that constructing task-specific knowledge can reduce the number of evaluations required for discovery, potentially lowering the cost of experimental investigation.

Effect of Knowledge Retrieval In our framework, knowledge retrieval plays a central role: knowledge retrieval determines which hypotheses guide subsequent actions and are later revisited in light of new observations. To examine its contribution, we ablate the LLM-based selection. As shown in Fig. 6, removing this lowers performance, suggesting that query-dependent selection helps identify useful knowledge for subsequent investigation, through which the selected hypotheses can be reassessed against new observations. Notably, removing LLM-based retrieval yields an AUC@10 of 0.364, comparable to 0.374 obtained without hypothesis revision in Fig. 4, suggesting that selecting relevant knowledge is important for realizing the benefits of hypothesis revision during optimization.

Qualitative Analysis We further qualitatively analyze the constructed hypotheses and principles to examine their reasonableness and scientific plausibility. As shown in Table 5, for pairs with the same labels, the hypotheses propose a possible explanation based on the shared structures. If the labels differ, they propose structural differences that may account for the contrasting outcomes. The induced principle provides a broader view by generalizing this explanation across related comparisons, identifying recurring motifs (i.e., epoxide) while distinguishing a specific structure (i.e., aminoalcohol) that lacks a relevant property (i.e., strained electrophile). Interestingly, this principle is consistent with prior studies linking epoxide-containing compounds to mutagenicity (Seiler, 1984; Rosman et al., 1988). This result shows how comparative hypotheses provide reasonable explanations for observed outcomes, while the induced principle captures scientifically plausible relationships across related molecules. An additional qualitative example of hypothesis revision is provided in Appendix E.

5 Conclusion

We studied the test-time construction of empirical principles for scientific discovery, focusing on how progressively accumulated observations can be transformed into knowledge that guides further investigation. To address this problem, we proposed PRINCIPIA, an LLM-based framework that forms comparative causal hypotheses, generalizes recurring relationships into inductive principles, and uses new observations to revise the hypotheses and reconstruct the principles. Experiments spanning molecular optimization, protein engineering, gene perturbation, and molecular property prediction demonstrate that PRINCIPIA outperforms baselines relying on internal knowledge, experience summaries, or external knowledge, with further analyses supporting its efficiency in reducing oracle evaluations and plausibility of the constructed principles. We believe our work highlights test-time hypotheses-to-principle construction as an important direction for advancing LLM-based scientific discovery, enabling accumulated observations to continually inform subsequent investigation.

AI USE STATEMENT

Generative AI tools were used in two roles in this work. First, large language models are integral components of PRINCIPIA and the experimental baselines. As described in the method, experimental sections, and Appendix, these models support comparative causal hypothesis generation, inductive principle construction, knowledge retrieval, hypothesis revision, and downstream prediction and candidate generation. Second, generative AI tools assisted with language editing, manuscript refinement, and code development. All research decisions, experimental design, analyses, reported results, and final manuscript content were reviewed and verified by the authors.

ETHICS STATEMENT

Our study exclusively utilizes publicly available datasets and models and does not involve human participants, personal data, or other sensitive information. All experiments are conducted using openly accessible resources and do not require access to private or restricted data. Given the nature and scope of our experiments, we do not anticipate any additional ethical concerns or risks associated with the methods presented in this work.

REPRODUCIBILITY STATEMENT

We provide an anonymous code repository containing our method as part of the submission. In addition, we provide detailed implementation specifications and algorithms for our method in Appendix Section A and Section D, experimental settings in Appendix Section B, and implementation details for the baselines in Appendix Section C.1 and Section C.2, while disclosing the prompts used for experiments in Appendix Section F.

REFERENCES

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.

Anthropic. Claude-mythos. https://www.anthropic.com/claude/mythos, 2026.

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. Researchagent: Iterative research idea generation over scientific literature with large language models. In *Proceedings of the 2025 conference of the nations of the Americas chapter of the association for computational linguistics: human language technologies (volume 1: long papers)*, pp. 6709–6738, 2025.

Viraj Bagal, Rishal Aggarwal, PK Vinod, and U Deva Priyakumar. Molgpt: molecular generation using a transformer-decoder model. *Journal of chemical information and modeling*, 62(9):2064–2076, 2021.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.

Paul Engelschalt, Maxime Röske, Johanna Penzlin, Dirk Krüger, and Annette Upmeier zu Belzen. Abductive reasoning in modeling biological phenomena as complex systems. *Frontiers in Education*, 8:1170967, 2023.

Wenhao Gao, Tianfan Fu, Jimeng Sun, and Connor Coley. Sample efficiency matters: a benchmark for practical molecular optimization. *Advances in neural information processing systems*, 35: 21342–21357, 2022.

Google DeepMind. Gemini 3.1 flash-lite. https://deepmind.google/models/model-cards/gemini-3-1-flash-lite/, 2026. Model Card.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, et al. Accelerating scientific discovery with co-scientist. *Nature*, pp. 1–3, 2026.

Jeff Guo and Philippe Schwaller. Augmented memory: Capitalizing on experience replay to accelerate de novo molecular design. *arXiv preprint arXiv:2305.16160*, 2023.

John H Holland. Outline for a logical theory of adaptive systems. *Journal of the ACM (JACM)*, 9(3): 297–314, 1962.

Jim Hone, V. Alistair Drake, and Charles J. Krebs. Prescriptive and empirical principles of applied ecology. *Environmental Reviews*, 23(2):170–176, 2015.

Jin Huang, Silviu Cucerzan, Sujay Kumar Jauhar, and Ryen W White. Idea2plan: Exploring ai-powered research planning. *arXiv preprint arXiv:2510.24891*, 2025.

Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

John J Irwin and Brian K Shoichet. Zinc- a free database of commercially available compounds for virtual screening. *Journal of chemical information and modeling*, 45(1):177–182, 2005.

Jan H Jensen. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572, 2019.

Jiachen Jiang, Tianyu Ding, and Zhihui Zhu. Deltaevolve: Accelerating scientific discovery through momentum-driven evolution. *arXiv preprint arXiv:2602.02919*, 2026.

Antonis Kakas and Loizos Michael. Abduction and argumentation for explainable machine learning: A position survey. *arXiv preprint arXiv:2010.12896*, 2020.

Evelina Lamma, Paola Mello, Fabrizio Riguzzi, Floriana Esposito, Stefano Ferilli, and Giovanni Semeraro. *Cooperation of Abduction and Induction in Logic Programming*, pp. 233–252. Springer Netherlands, Dordrecht, 2000.

Robert Lange, Yuki Imajuku, and Edoardo Cetin. Shinkaevolve: Towards open-ended and sample-efficient program evolution. In *International Conference on Learning Representations*, volume 2026, pp. 74026–74078, 2026.

Seul Lee, Jaehyeong Jo, and Sung Ju Hwang. Exploring chemical space with score-based out-of-distribution generation. In *International Conference on Machine Learning*, pp. 18872–18892. PMLR, 2023.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Long Li, Weiwen Xu, Jiayan Guo, Ruochen Zhao, Xingxuan Li, Yuqian Yuan, Boqiang Zhang, Yuming Jiang, Yifei Xin, Ronghao Dang, et al. Chain of ideas: Revolutionizing research via novel idea development with llm agents. In *EMNLP (Findings)*, pp. 8971–9004, 2025.

Gang Liu, Yihan Zhu, Jie Chen, and Meng Jiang. Scientific algorithm discovery by augmenting alphaevolve with deep research. *arXiv preprint arXiv:2510.06056*, 2025.

Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of ai research. *Nature*, 651(8107):914–919, 2026.

Sichun Luo, Yi Huang, Haochen Luo, Fengyuan Liu, Guanzhi Deng, Lei Li, Qinghua Yao, Zefa Hu, Junlan Feng, and Qi Liu. Seaevo: Advancing algorithm discovery with strategy space evolution. *arXiv preprint arXiv:2604.24372*, 2026.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature machine intelligence*, 6(5):525–535, 2024.

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.

Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):48, 2017.

OpenAI. openai-text-embedding-3-large. https://openai.com/index/new-embedding-models-and-api-updates/, 2024.

OpenAI. openai-gpt56. https://openai.com/index/gpt-5-6/, 2026a.

OpenAI. openai-gpt-rosalind. https://openai.com/index/introducing-gpt-rosalind/, 2026b.

Robert A. Pearlstein, Hongbin Wan, and Vibhas Aravamuthan. Toward in vivo relevant drug design. *Drug Discovery Today*, 26(3):637–650, 2021.

Yingming Pu, Tao Lin, and Hongyu Chen. Piflow: Principle-aware scientific discovery with multi-agent collaboration. *arXiv preprint arXiv:2505.15047*, 2025.

Nian Ran, Yue Wang, Xiaoyuan Zhang, Zhongzheng Li, Qingsong Ran, Wenhao Li, and Richard Allmendinger. Exllm: Experience-enhanced llm optimization for molecular design and beyond. *arXiv preprint arXiv:2502.12845*, 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 *International Conference on Learning Representations*, volume 2025, pp. 26417–26466, 2025.

L. B. Rosman, P. K. Chakraborty, E. A. Messerly, and J. E. Sinsheimer. Mutagenicity of aromatic glycidyl ethers with Salmonella. *Mutation Research*, 206(1):115–125, 1988.

Samantha G Scharenberg, Wentao Dong, Ali Ghoochani, Kwamina Nyame, Roni Levin-Konigsberg, Aswini R Krishnan, Eshaan S Rawat, Kaitlyn Spees, Michael C Bassik, and Monther Abu-Remaileh. An spns1-dependent lysosomal lipid transport pathway that enables cell survival under choline limitation. *Science Advances*, 9(16):eadf8966, 2023.

Samuel M. Scheiner and Michael R. Willig. A general theory of ecology. *Theoretical Ecology*, 1(1): 21–28, 2008.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using llm agents as research assistants. *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 5977–6043, 2025.

Samuel Schmidgall, Xiaokai Zhu, Marian Shaw, Lin Yang, Valentin Liévin, Jingyun Yang, Yuchen Zhuang, Tim Strother, Alex Bijamov, Min Woo Sun, Anil Palepu, Justin Chen, David Steiner, Jacqueline Shreibati, Wei-Hung Weng, Yilin Zhao, Xingjian Hu, Nicholas Zahn, Sadhya Garg, Julia Kirby, Yuxiang Gan, Jiaoli Li, Divy Thakkar, Shekoofeh Azizi, David Racz, Juraj Gottweis, Vivek Natarajan, Chenglin Wu, Tal Danino, Keran Rong, Haozhe Wang, Benoit Schillings, Yong Cheng, Quoc V. Le, and Tao Tu. Accelerating scientific research with gemini in the real-world, 2026.

Ralf Schmidt, Zachary Steinhart, Madeline Layeghi, Jacob W Freimer, Raymund Bueno, Vinh Q Nguyen, Franziska Blaeschke, Chun Jimmie Ye, and Alexander Marson. Crispr activation and interference screens decode stimulation responses in primary human t cells. *Science*, 375(6580): eabj4008, 2022.

J. P. Seiler. The mutagenicity of mono- and di-functional aromatic glycidyl compounds. *Mutation Research*, 135(3):159–167, 1984.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Zhangde Song, Jieyu Lu, Yuanqi Du, Botao Yu, Thomas M Pruyn, Yue Huang, Kehan Guo, Xiuzhe Luo, Yuanhao Qu, Yi Qu, et al. Evaluating large language models in scientific discovery. *arXiv preprint arXiv:2512.15567*, 2025.

Rosni Vasu, Chandrayee Basu, Bhavana Dalvi, Cristina Sarasua, Peter Clark, and Abraham Bernstein. Hyper: Literature-grounded hypothesis generation and distillation with provenance. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25424–25449, 2025.

Haorui Wang, Marta Skreta, Cher-Tian Ser, Wenhao Gao, Lingkai Kong, Felix Strieth-Kalthoff, Chenru Duan, Yuchen Zhuang, Yue Yu, Yanqiao Zhu, et al. Efficient evolutionary search over chemical space with large language models. In *International Conference on Learning Representations*, volume 2025, pp. 51694–51727, 2025a.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. Scimon: Scientific inspiration machines optimized for novelty. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 279–299, 2024.

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*, 2025b.

Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

Peter Willett, John M Barnard, and Geoffrey M Downs. Chemical similarity searching. *Journal of chemical information and computer sciences*, 38(6):983–996, 1998.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

Xinjie Yao, Xingxin Xu, Xiyuan Gao, Zhoupeng Guo, Kunlong Yang, Dengyu Zhao, Siqi Zhao, Zhihe Fan, Yichen Dong, Xin Li, et al. Towards a new paradigm of scientific discovery with socialized artificial intelligence. *arXiv preprint arXiv:2608.02775*, 2026.

Mert Yuksekgonul, Daniel Koceja, Xinhao Li, Federico Bianchi, Jed McCaleb, Xiaolong Wang, Jan Kautz, Yejin Choi, James Zou, Carlos Guestrin, et al. Learning to discover at test time. *arXiv preprint arXiv:2601.16175*, 2026.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026a.

Yikun Zhang, Xiwei Cheng, Tianyu Liu, Yuanqi Du, and Wengong Jin. Drugsage: Self-evolving agent experience for efficient state-of-the-art drug discovery. *arXiv preprint arXiv:2605.15461*, 2026b.

Appendix

A Implementation Details of Principia

Comparative Evidence and Hypothesis Generation For scientific optimization, we construct comparative evidence from the source inputs and evaluated candidates recorded during candidate generation. Each comparison contains the two inputs and their oracle scores, allowing the hypothesis generator to explain the observed score difference in the context of the task. For molecular property prediction, we pair each newly observed molecule with up to three reference molecules ranked by Tanimoto similarity (Willett et al., 1998), excluding the molecule itself. The reference pool contains the molecules observed in previous rounds and those whose labels have become available in the current round. Reference selection does not filter by label, so comparisons can involve either matching or differing outcomes. We generate hypotheses separately for each reference-target comparison and retain both inputs and their observations. Molecular comparisons use SMILES (Weininger, 1988) representations, protein comparisons use amino-acid sequences, and gene comparisons use HGNC[1] gene symbols. The task-specific hypothesis-generation prompts are provided in Figures 10, 17, and 24 in Appendix Section F.

Clustering and Principle Construction We group inputs using soft-balanced k-medoids with a domain-specific pairwise distance. For molecules, we compute Tanimoto distance from 2,048-bit Morgan fingerprints with radius 2. For protein sequences, we use normalized Hamming distance for equal-length sequences and Jaccard distance between sets of overlapping amino-acid 3-mers for unequal-length sequences. For genes, we embed the gene symbols using `text-embedding-3-large` (OpenAI, 2024), normalize the embeddings to unit length, and use cosine distance. In optimization, clustering is performed on the current population, while prediction groups the molecules observed so far. For n inputs, we set the number of clusters to $\max(1, \lfloor n/10 \rfloor)$, targeting ten inputs per cluster. We use a size-balancing penalty of 0.01, minimum and maximum cluster sizes of 5 and 15. For each cluster, we collect the comparative evidence and associated hypotheses linked to its members. An LLM then summarizes the recurring relationships and conflicting observations into a principle, retaining the association with the cluster members. The task-specific principle construction prompts are provided in Figures 11, 18, and 25 in Appendix Section F.

Knowledge Retrieval We retrieve hypotheses and principles independently through candidate selection followed by LLM selection. We first a construct hypothesis candidate pool. For hypothesis retrieval in optimization, we rank comparison records by their fitness changes. We then alternate between the largest improvements and declines, selecting up to 100 unique records. For prediction, we rank comparison records by the Tanimoto similarity between each pair's target molecule and the query. We then select up to 50 pairs with matching labels and 50 pairs with different labels from the ranked records. The LLM receives the candidate records, query input, and task description and selects relevant comparisons. To construct the principle candidate pool, we rank clusters by their maximum member similarity to the query using the domain-specific similarity above, retain at most 100 clusters, and ask the LLM to select relevant principles. The optimization configuration retrieves up to three hypothesis records and one principle per query. In prediction, the hypothesis retrieval count is configurable, while the LLM determines how many principles to select. The selected hypotheses and principles are provided as textual context for candidate generation or property prediction. The task-specific prompts for causal hypothesis and inductive principle retrieval are provided in Figures 12, 13, 19, and 20 in Appendix Section F.

Hypothesis Revision via New Observations We revise the hypotheses retrieved and used during the current generation or prediction round. We uniformly sample up to five new observations. In optimization, these observations contain source-target inputs and their evaluated scores. In prediction, they contain a probe molecule, its predicted label, and its observed label. The LLM receives the original comparative evidence, its current hypotheses, the sampled observations, and the task description, and can retain, revise, remove, or add hypotheses. The updated hypotheses replace

[1] https://www.genenames.org/

the corresponding memory records. The task-specific hypothesis-revision prompts are provided in Figures 14, 21, and 28 in Appendix Section F.

B Experimental Settings

B.1 Settings

Models We primarily use GPT-5-mini-2025-08-07 (Singh et al., 2025) for the optimization tasks, while using gpt-5.4-mini-2026-03-17 as the backbone model for molecular property prediction. For both optimization and prediction tasks, we consistently set the temperature to 1.0 as these models do not allow to use custom temperature.

B.2 Scientific Optimization

Molecular Optimization We evaluate five challenging tasks from the Practical Molecule Optimization (PMO) benchmark (Gao et al., 2022): JNK3, GSK3β, Median1, Median2, and Valsartan SMARTS. JNK3 and GSK3β reward predicted inhibitory activity against their respective targets. Median1 and Median2 measure the simultaneous similarity to two reference molecules: camphor and menthol for Median1, and tadalafil and sildenafil for Median2. Valsartan SMARTS combines a target-substructure requirement with physicochemical property objectives. We use the benchmark scoring functions as experimental oracles, where higher scores indicate better satisfaction of the corresponding objectives. For tasks whose descriptions and objectives are provided in Wang et al. (2025a), we adopt their specifications. For the remaining task, we reimplement the task description following Ran et al. (2025), matching the style of its instructions with Wang et al. (2025a) (Figure 16). Each run is initialized with a pre-sampled population of 200 molecules randomly drawn from the ZINC (Irwin & Shoichet, 2005) pool and generates up to 100 offspring per generation. We use the same initial population across corresponding methods and set the nominal evaluation budget to 1,000 distinct molecules (oracle calls), including the initial population.

Protein Sequence Engineering We evaluate Syn-3bfo, AAV, and GB1 using the task-specific fitness oracles provided by the protein optimization benchmark (Wang et al., 2025b). Each task represents a distinct protein property and uses a corresponding evaluation protocol. We therefore select three tasks to evaluate the performance across different protein properties and evaluation settings. Syn-3bfo uses a synthetic fitness landscape derived from a Potts model, AAV uses a pretrained convolutional predictor of capsid fitness, and GB1 uses experimentally measured fitness values through a sequence lookup table. We follow the corresponding oracle implementations for score normalization. The search operates on amino-acid sequences of the task-specific length using the 20 standard amino acids. Each optimization task focuses on maximizing its respective fitness objective. We reimplemented the task descriptions based on each optimization objective, due to the absence of task-specific optimization prompts available in the original benchmark (Figure 23). Each run starts from 48 pre-sampled sequences, including the wild-type sequence, and generates up to 48 offspring per generation. The initial population is shared across the corresponding methods, and the nominal evaluation budget is 200 distinct sequences (oracle calls), including initialization.

Gene Perturbation We evaluate Scharenberg22 and IFNG using the CRISPR-screen datasets and task definitions adopted by BioDiscoveryAgent (Roohani et al., 2025). Inputs are gene symbols, and the oracle returns a score derived from the recorded perturbation effect. Scharenberg22 concerns lysosomal choline recycling (Scharenberg et al., 2023), and the optimization maximizes the recorded screen score. IFNG concerns regulation of interferon-γ production in human T cells. Its oracle uses the magnitude of the recorded effect, so both strong increases and strong decreases receive high scores. We reuse the task descriptions opened in the original benchmark for each task Roohani et al. (2025) (Figure 30). Since our optimization objectives are scalar-valued, we maximize the raw fitness score for Scharenberg22, while optimizing the absolute value of the raw fitness score for IFNG, as its task objective is to measure the magnitude of the recorded effect. The two tasks differ in the size of their respective search spaces, with the latter involving a larger search space than the former, allowing us to examine the effect of Principia across tasks with different search-space sizes.

We use pre-sampled initial gene populations of 32 for Scharenberg22 and 128 for IFNG, with matching offspring sizes. The corresponding nominal evaluation budgets in the current execution scripts are 192 and 768 distinct genes (oracle calls), ensuring that 160 and 640 novel genes are proposed, following the evaluation setting from Roohani et al. (2025). Ground-truth hit genes are obtained from the benchmark's provided hit sets, and are measured among distinct K (160, 640) proposed genes commonly across baselines. Furthermore, due to the high frequency of duplicate gene generation, we explicitly provide the backbone LLM operator with previously generated candidates across all baselines (MolLEO, PiFlow, ExLLM, DeepEvolve, and WebSearch) to promote exploration of new candidates. For PiFlow, we additionally provide a list of previously generated invalid genes to reduce repeated invalid generations and improve the efficiency of candidate generation, given its tendency to repeatedly produce invalid candidates.

Evaluation Protocol For optimization, AUC@K summarizes how quickly a method discovers high-scoring inputs, a representative metric for assessing the performance of scientific discovery frameworks, measuring its area under the curve of top-k average fitness values of discovered candidates. At successive evaluation counts, we compute the mean score of the ten best inputs observed so far, integrate this trajectory using the trapezoidal rule, and normalize by the evaluation budget, following Gao et al. (2022); Wang et al. (2025a), which we use summation frequency of 100 oracle calls (distinct candidates). Higher AUC@K therefore reflects both the quality of discovered inputs and the speed at which they are found. Avg@K for proteins denotes the mean scores of the best K evaluated inputs, respectively. For gene perturbation, Hit Ratio denotes the fraction of benchmark hit genes recovered among the genes proposed during search; this is a recall measure, distinct from the fraction of proposed genes that are hits. The implementation supports both essential-gene-filtered (Table 7) and unfiltered variants (Table 2) and excludes the random initial population from proposal-based gene metrics. The optimization budget counts distinct evaluated inputs, with repeated inputs reusing recorded scores and initial population evaluations contributing to the count. All results are macro-averaged across three random seeds.

B.3 Molecular Property Prediction

Datasets and Data Preparation We evaluate eight binary classification tasks from Therapeutics Data Commons Huang et al. (2021). The toxicity-related tasks are DILI, hERG, Ames, and Skin Reaction, which concern liver injury, hERG channel blockade, mutagenicity, and skin reaction, respectively. The remaining tasks predict inhibition of CYP1A2, CYP2C9, CYP2D6, and CYP3A4. We prepare training, validation, and test partitions using scaffold splitting. After splitting, we randomly sample up to 200 training molecules and 50 test molecules per class for each run.

Sequential Knowledge Construction We divide the sampled training molecules into six equal rounds. Each training molecule appears in one round. The first round provides labeled observations for initializing the hypothesis and principle memories. In each subsequent round, the existing memories guide predictions for the newly introduced molecules without exposing their labels to the prediction prompts. Their labels are then revealed to provide new observations for hypothesis revision. We subsequently construct comparative hypotheses for the newly labeled molecules using references available by that round and rebuild the principles from the accumulated knowledge. After the sixth round, the resulting memories are fixed for held-out test prediction. The validation partition is not used by this sequential construction procedure, and test labels are excluded from hypothesis generation, revision, and principle construction.

Evaluation Protocol We evaluate the fixed memories on the held-out test molecules using accuracy and macro-F1. Accuracy is the proportion of test molecules whose predicted labels match the observed labels. Macro-F1 is the unweighted mean of the class-specific F1 scores, where each class-specific F1 combines precision and recall through their harmonic mean. Thus, both classes contribute equally to macro-F1 even when their sample counts differ. Responses from which no valid task label can be extracted are treated as incorrect predictions and remain in the evaluation set. The result aggregation computes the mean of the three different runs.

C BASELINES

C.1 SCIENTIFIC OPTIMIZATION

REINVENT We restrict our evaluation of the reinforcement-learning-based REINVENT algorithm to the pre-implemented code from the open-source official PMO (Gao et al., 2022) repository[2]. We adopt the default hyperparameters listed in the official code repository. Specifically, the agent policy is optimized with a learning rate of 0.0005 and a batch size of 64 molecules sampled per update. The reward-scaling coefficient sigma is set to 500, and the experience-replay parameter is set to 24, specifying the number of previously seen high-scoring molecules that are re-sampled from a prioritized memory and added to each update to reinforce promising candidates. We set the maximum number of oracle calls to 1000 and the patience to 10, with patience threshold of 1e-3, and AUC computation frequency value to 100, matching the experimental setting of PRINCIPIA and its LLM-based discovery baselines.

Augmented Memory We additionally evaluate optimization algorithms with Augmented Memory, a reinforced extension of REINVENT that introduces memory augmentation. Our implementation follows the official PMO repository. Consistent with the REINVENT protocol, we cap the maximum number of oracle calls at 1,000 and use a patience of 10 with a threshold of 1e-3, evaluating AUC every 100 oracle calls. As with REINVENT, we retain the default hyperparameters hard-coded in the official repository: a batch size of 64, sigma of 500, replay-buffer size of 1,000, learning rate of 5e-4, and 2 augmentation rounds per augmented-memory update.

Graph GA Likewise, as with other machine-learning-based optimization algorithms, we follow the implementation details provided in the official PMO repository. We set the maximum number of oracle calls to 1,000, the patience to 10, with patience threshold of 1e-3, and the AUC evaluation frequency to 100, consistent with the settings used by other machine-learning-based molecular optimization methods. Since Graph GA is a population-based evolutionary search method, we use the same pre-sampled initial population as PRINCIPIA and the other baselines, consisting of 200 samples. We additionally set the offspring population size to 100 to align its hyperparameter settings with those used for PRINCIPIA. For the mutation rate, we follow the default configuration in the official repository and set it to 0.067.

MolLEO We reimplement MolLEO based on the original implementation (Wang et al., 2025a)[3] and the reimplementation of Song et al. (2025)[4]. Our implementation provides the common architecture for PRINCIPIA, ExLLM, WebSearch, and DeepEvolve, following the original GPT-4-based design. We support both original architectures, joint mutation and crossover-then-mutate, and use the latter by default as it approximately halves LLM calls. For gene perturbation, we use joint mutation since crossover is not directly applicable. We adopt the mating-pool strategy of Song et al. (2025), setting its size to the number of offspring per generation for faster convergence under limited oracle budget, and normalizing fitness values for mating pool selection to handle negative scores. At each reproduction step, the LLM generates multiple candidates, and we select the first valid, previously unevaluated candidate, falling back to the first valid candidate if all proposals are duplicates. This reduces redundant oracle queries under a limited evaluation budget. All other hyperparameters, including oracle-call budget, population settings, patience, and convergence threshold, are matched to PRINCIPIA.

ExLLM Following the experience-guided memory mechanism of ExLLM (Ran et al., 2025) implemented in the official repository[5], our textual-memory baseline augments the MolLEO baseline with an experience summarization mechanism. Once per generation, after more than 100 candidates have been evaluated, an LLM analyzes 10 sampled high-performing and 10 low-performing candidates from top-100 candidates based on scalar fitness to distill a rolling, 200-word natural-language experience summary. The newly generated summary is integrated with the previous summary and injected into each generation-level LLM call with probability 0.5, using ExLLM's

[2] https://github.com/wenhao-gao/mol_opt.git
[3] https://github.com/zoom-wang112358/MOLLEO
[4] https://github.com/HowieHwong/sde-harness
[5] https://github.com/HuskyNian/ExLLM

experience-injection wrapper and analysis instructions. Since ExLLM's base implementation targets a multi-objective optimization setting, whereas our evaluation optimizes a single scalar objective, we adapt its candidate-summarization procedure to account for this mismatch in the optimization setting. Specifically, we describe each candidate using its scalar fitness value together with a task-specific description prompt, rather than an explicit listing of multiple objective values. All remaining optimization hyperparameters are kept identical to those used by PRINCIPIA.

PiFlow Since no official implementation is publicly available, we reimplement PiFlow (Pu et al., 2025) following the original paper. We retain its Min-Max principle scoring, score-based Refine/Validate/Explore procedure, and round-robin group-chat protocol, using the default hyperparameters $\lambda = 0.5$, $K = 3$, and $M = 10$ without tuning. We minimally adapt the Planner and Hypothesis prompts to our experimental settings, replace the Experiment agent with a deterministic oracle, and use the same backbone LLM as PRINCIPIA. We use `text-embedding-3-large` (OpenAI, 2024) for semantic novelty and initialize three principles using the paper's cold-start procedure, due to the absence of pre-configured principles in our experimental setting. At each LLM call, we generate three candidates and evaluate the first valid candidate without explicit duplicate checking, matching original paper's intent for generating single candidate based on single hypothesis. For molecule optimization and protein sequence engineering, we follow the paper's 24-iteration short-horizon protocol, including the cold-start iteration, and evaluate top-k fitness and AUC over de-duplicated candidates. For gene perturbation, we instead match the oracle-call budgets of PRINCIPIA and other baselines (768 and 192) and compute the metric (hit-ratio) over the resulting distinct candidates.

DeepEvolve We adapt DeepEvolve's deep-research agent (Liu et al., 2025) and official implementation[6] as a retrieval-augmented baseline. Since candidate generation is already handled by our MolLEO architecture, we retain only DeepEvolve's query-planning and web-search summarization components, injecting the resulting summaries into the design prompts. For each selected parent candidate (or pair), the planner generates $K = 4$ queries based on the task objective, candidate(s) and oracle scores, evolutionary history, and top-$N = 5$ scoring candidates; the summarizer executes the queries and concatenates the results, with exclusive access to the search tool. We preserve the original prompt structure while replacing its generation-based progress signal with the fraction of the oracle-call budget consumed and retaining the 0.5 early/mature threshold. We reconstruct candidate ancestry chronologically and use the top-N candidates as $(\text{candidate}, \text{score})$ inspiration pairs. To limit information leakage, queries are restricted to general scientific principles rather than specific candidate solutions. The planner and summarizer use fixed models independent of the optimization backbone, while GPT-5-mini (Singh et al., 2025) is used as the optimization backbone, with all other optimizer hyperparameters matched to PRINCIPIA.

WebSearch As a web-based retrieval baseline, we use Google Search via the Serper API[7]. For each parent candidate (or parent pair) selected for an LLM operation, a planner generates a search query conditioned on the target property, task objective, parent candidate(s) and oracle scores, and current optimization progress. The query is issued to the search API, and the top-k retrieved passages are directly injected into the design prompt without LLM-based summarization. We adapt the query-planning prompt from our DeepEvolve implementation by removing the evolutionary-history and inspiration blocks and instructing the planner to generate concise, keyword-style queries of 5–8 words for general scientific search. To align with PRINCIPIA, we fix the number of injected passages to 4, measure progress as the fraction of the oracle-call budget consumed, and use GPT-5-mini (Singh et al., 2025) as the fixed planner. The planner is constrained to retrieve general scientific principles rather than information about specific candidates to reduce potential information leakage and align with our research objectives (scientific principles), while all other optimization hyperparameters, including the initial population and population size, are matched to PRINCIPIA.

C.2 MOLECULAR PROPERTY PREDICTION

Direct Prediction The LLM predicts the molecular property from the task description and the query molecule represented as SMILES. The prompt specifies the required answer format without providing labeled examples or additional scientific knowledge.

[6] https://github.com/liugangcode/deepevolve.git
[7] https://serper.dev/

Few-Shot Prediction We provide $k \in \{1, 3, 5, 7\}$ labeled training molecules as examples for each query. Examples are selected by their Tanimoto similarity to the query using 2,048-bit Morgan fingerprints with radius 2. Each example contains the molecular SMILES and its observed label, and the selected examples are included before the query in the prediction prompt.

Internal Principles We augment the prediction prompt with a fixed set of task-specific scientific principles generated by the internal knowledge of LLMs. These principles describe molecular characteristics and physicochemical factors relevant to the target property. The same principles are provided for every query within a task and remain fixed throughout evaluation, without being updated from the observed training labels.

External Principles We obtain task-specific scientific principles through web search. Given the task description, an LLM formulates search queries targeting general mechanisms, structure-activity relationships, and physicochemical determinants. We issue these queries through Google Search via the Serper API and collect result snippets together with page content from the top-ranked sources. An LLM then summarizes the retrieved material into predictive principles, with instructions to include only claims supported by the sources. Search queries target general domain knowledge and exclude specific query molecules, benchmark names, and lists of labeled compounds. The resulting principles are stored and provided as the same textual context for all query molecules within the task.

Textual Summary We construct a single textual summary by sequentially processing the batched training molecules. At each step, the LLM receives the task description, the previous summary, and the new molecular SMILES-label pairs. It updates the summary to capture label-discriminative patterns, structural alerts, recurring molecular features, and exceptions supported by the observations. After all training batches have been processed, the final summary is fixed and included in the prediction prompt for every test molecule. This baseline accumulates guidance directly in a single text without maintaining separate comparative hypotheses and inductive principles.

D ALGORITHMS

In this section, we present the algorithmic details of PRINCIPIA. Algorithm 1 summarizes the overall workflow, which consists of three stages: (1) forming causal hypotheses with comparative evidence (Section 3.1), (2) applying and refining comparative hypotheses based on experimental outcomes (Section 3.3), and (3) inductively constructing scientific principles (Section 3.2). We describe the shared procedures for optimization and prediction tasks in Algorithm 2, and the task-specific procedures in Algorithms 3 and 4, respectively.

Algorithm 1 PRINCIPIA- Main Workflow

Require: Task description q; records $\mathcal{D} = \{(x_i, \mathcal{O}(x_i))\}_{i=1}^{N}$; oracle $\mathcal{O}$; distance $d(\cdot,\cdot)$; reference neighbors k_{ref}; cluster size k_c; retrieval sizes k_h, k_p; revision evidence size k_r; rounds T; oracle-call budget B and offspring size M (optimization)
Ensure: Hypothesis memory $\mathcal{M}_h$, principle memory $\mathcal{M}_p$

$\mathcal{M}_h \leftarrow \emptyset$; $\mathcal{M}_p \leftarrow \emptyset$
if task is optimization **then**
  $\mathcal{P} \leftarrow \{x \mid (x, \mathcal{O}(x)) \in \mathcal{D}\}$ ▷ Initial population of size N
else
  Partition $\mathcal{D}$ into T slices $\{\mathcal{B}_t\}_{t=1}^{T}$; $\mathcal{P} \leftarrow \emptyset$ ▷ Seeded round-robin; observed set grows per round
for $t = 1$ **to** T **do**
  if $t = 1$ **then**
    $(\mathcal{C}, \sim) \leftarrow$ FORMCOMPARATIVEEVIDENCE$(\mathcal{D})$ ▷ Section 3.1
  else
    $(\mathcal{C}, \mathcal{B}, \mathcal{U}) \leftarrow$ APPLYHYPOTHESES$(\mathcal{M}_h, \mathcal{M}_p)$ ▷ Section 3.3
    $\mathcal{M}_h \leftarrow$ REVISEHYPOTHESES$(\mathcal{M}_h, \mathcal{U}, \mathcal{B})$ ▷ Section 3.3
  $\mathcal{M}_h \leftarrow$ GENERATEHYPOTHESES$(\mathcal{M}_h, \mathcal{C})$ ▷ Section 3.1
  $\mathcal{M}_p \leftarrow$ INDUCEPRINCIPLES$(\mathcal{M}_h, \mathcal{D})$ ▷ Section 3.2
  if task is optimization **then**
    if $|\mathcal{D}| \geq B$ **then**
      break ▷ Oracle-call budget exhausted
  else
    continue ▷ Prediction: terminates after T rounds
return $\mathcal{M}_h, \mathcal{M}_p$

Algorithm 2 PRINCIPIA - Shared procedures (task-agnostic)

procedure REVISEHYPOTHESES$(\mathcal{M}_h, \mathcal{U}, \mathcal{B})$
  for each $(e_{i\to j}, H_{i\to j}) \in \mathcal{U}$ **do**
    $H_{i\to j} \leftarrow \text{LLM}_{\text{refine}}\big(e_{i\to j}, H_{i\to j}, \text{SAMPLE}(\mathcal{B}, k_r); q\big)$ ▷ $e_{i\to j}$ remains attached
  return $\mathcal{M}_h$

procedure GENERATEHYPOTHESES$(\mathcal{M}_h, \mathcal{C})$
  for each $e_{i\to j} \in \mathcal{C}$ **do**
    Store $\big(e_{i\to j}, \text{LLM}_{\text{hypothesis}}(e_{i\to j}; q)\big)$ in $\mathcal{M}_h$ as directed edge $x_i \to x_j$
  return $\mathcal{M}_h$

procedure INDUCEPRINCIPLES$(\mathcal{M}_h, \mathcal{D})$
  $\{G_k\}_{k=1}^{K} \leftarrow k\text{-MEDOIDS}\big(\mathcal{P}, d, \lfloor |\mathcal{P}|/k_c \rfloor\big)$
  $\phi \leftarrow (\, i \in G_k \vee j \in G_k \,)$ if task is optimization else $(\, j \in G_k \,)$ ▷ Prediction assigns each edge to its target's cluster
  for $k = 1$ **to** K **do**
    $P_k \leftarrow \text{LLM}_{\text{principle}}\big(\{(e_{i\to j}, H_{i\to j}) \mid (i,j) \in \mathcal{A}_h,\ \phi\}; q\big)$
  return $\mathcal{M}_p = \{(G_k, P_k)\}_{k=1}^{K}$

Algorithm 3 PRINCIPIA - Optimization-specific procedures

procedure FORMCOMPARATIVEEVIDENCE($\mathcal{D}, \mathcal{M}_h{=}\emptyset, \mathcal{M}_p{=}\emptyset$)
  $\mathcal{S}, \Pi, \mathcal{U} \leftarrow \emptyset$; $\mathcal{Q} \leftarrow \textsc{Select}(\mathcal{P}, M)$ ▷ Fitness-proportionate
  if joint **then** ▷ For Gene Perturbation
    $\mathcal{X} \leftarrow \mathcal{Q}$ ▷ No crossover; parents are mutated directly
  else
    $\mathcal{X} \leftarrow \{\textsc{Crossover}(x_1, x_2) \mid (x_1, x_2) \in \mathcal{Q}\}$; $\mathcal{S} \leftarrow \mathcal{X}$
    $\Pi \leftarrow \{x_1 \to x_c,\ x_2 \to x_c \mid x_c{=}\textsc{Crossover}(x_1, x_2), (x_1, x_2) \in \mathcal{Q}\}$
  for each $x_c \in \mathcal{X}$ **do**
    $\mathcal{R}_h \leftarrow \mathrm{LLM}_{\text{select}}(x_c, \mathcal{M}_h, k_h; q)$
    $\mathcal{R}_p \leftarrow \mathrm{LLM}_{\text{select}}(x_c, \mathcal{M}_p, k_p; q)$ ▷ $\emptyset$ if no knowledge
    $x_m \leftarrow \mathrm{LLM}_{\text{act}}(x_c, \mathcal{R}_h, \mathcal{R}_p; q)$; $\mathcal{U} \leftarrow \mathcal{U} \cup \mathcal{R}_h$
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{x_m\}$; $\Pi \leftarrow \Pi \cup \{x_c \to x_m\}$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x, \mathcal{O}(x)) \mid x \in \mathcal{S}\}$; $\mathcal{P} \leftarrow \mathrm{Top}_N(\mathcal{P} \cup \mathcal{S})$ ▷ Survivor selection by $\mathcal{O}(\cdot)$
  return $\big(\{e_{i \to j} \mid x_i \to x_j \in \Pi\},\ \mathcal{U}\big)$

procedure APPLYHYPOTHESES($\mathcal{M}_h, \mathcal{M}_p$) ▷ Knowledge-guided generation
  $(\mathcal{C}, \mathcal{U}) \leftarrow \textsc{FormComparativeEvidence}(\mathcal{D}, \mathcal{M}_h, \mathcal{M}_p)$
  return $(\mathcal{C}, \mathcal{C}, \mathcal{U})$ ▷ New observations are also new evidence

function $\mathrm{LLM}_{\text{ACT}}(x, \mathcal{R}_h, \mathcal{R}_p; q)$
  return $\mathrm{LLM}_{\text{mutate}}\big(x, \mathcal{O}(x), \mathcal{R}_h, \mathcal{R}_p; q\big)$ ▷ Knowledge-guided experimental action

Algorithm 4 PRINCIPIA - Prediction-specific procedures

procedure FORMCOMPARATIVEEVIDENCE($\mathcal{D}, \mathcal{M}_h{=}\emptyset, \mathcal{M}_p{=}\emptyset$)
  $\mathcal{C} \leftarrow \emptyset$; $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{B}_t$ ▷ Next slice $\mathcal{B}_t$ becomes observed (no oracle, no selection)
  for each $x_j \in \mathcal{B}_t$ **do**
    $\mathcal{N}_j \leftarrow \mathrm{kNN}_d\big(x_j, \mathcal{P} \setminus \{x_j\}, k_{\text{ref}}\big)$ ▷ Nearest observed references; labels not filtered
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{e_{i \to j} \mid x_i \in \mathcal{N}_j\}$ ▷ $\mathcal{O}(x_i), \mathcal{O}(x_j), [\mathcal{O}(x_i) \neq \mathcal{O}(x_j)]$ attached
  return $(\mathcal{C},\ \emptyset)$ ▷ $\mathcal{U}{=}\emptyset$: hypotheses are applied in APPLYHYPOTHESES

procedure APPLYHYPOTHESES($\mathcal{M}_h, \mathcal{M}_p$) ▷ Knowledge-guided prediction on the unseen slice $\mathcal{B}_t$
  $\mathcal{B}, \mathcal{U} \leftarrow \emptyset$
  for each $x \in \mathcal{B}_t$ **do** ▷ Predict *before* $\mathcal{P}$ grows, so memory holds only prior rounds
    $\mathcal{R}_h \leftarrow \mathrm{LLM}_{\text{select}}(x, \mathcal{M}_h, k_h; q)$
    $\mathcal{R}_p \leftarrow \mathrm{LLM}_{\text{select}}(x, \mathcal{M}_p, k_p; q)$ ▷ $\emptyset$ if no principles, k_p for fallback, no principle number limit
    $\hat{y} \leftarrow \mathrm{LLM}_{\text{act}}(x, \mathcal{R}_h, \mathcal{R}_p; q)$
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{(x, \mathcal{O}(x), \hat{y})\}$; $\mathcal{U} \leftarrow \mathcal{U} \cup \mathcal{R}_h$
  $(\mathcal{C}, \sim) \leftarrow \textsc{FormComparativeEvidence}(\mathcal{D})$ ▷ Form edges for $\mathcal{B}_t$ and grow $\mathcal{P}$
  return $(\mathcal{C}, \mathcal{B}, \mathcal{U})$ ▷ $\mathcal{B}$: prediction outcomes, distinct from $\mathcal{C}$

function $\mathrm{LLM}_{\text{ACT}}(x, \mathcal{R}_h, \mathcal{R}_p; q)$
  return $\mathrm{LLM}_{\text{predict}}\big(x, \mathcal{R}_h, \mathcal{R}_p; q\big)$ ▷ No oracle mutation; label prediction

Table 6: Importance of diversifying learning signals for hypothesis retrieval in Scientific Discovery on GSK3β optimization, using `gpt-5-mini` as backbone models.

Method	AUC@10 ↑	Δ ↑
None	0.341	—
Δ Random	0.366	+0.025
Δ Worst	0.379	+0.038
Δ Best	0.372	+0.031
Δ Mixture (Ours)	**0.452**	+0.111

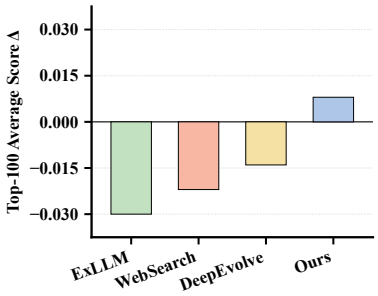


Figure 7: Analysis on multi-objective Syn-3bfo optimization. We report the performance Δ of each variant relative to plain optimization.

Table 7: Analysis of the performance on the Scharenberg22 and IFNG tasks using an additional metric, Hit Ratio w/o Essential, using `gpt-5-mini` as backbone models.

Variant	Scharenberg22	IFNG
MolLEO*	0.265	0.114
ExLLM*	0.286	0.117
PiFlow*	0.272	0.118
WebSearch*	0.218	0.117
DeepEvolve*	0.265	0.113
Ours	**0.320**	**0.123**

E Supplementary Experiments

Importance of Diversifying Learning Signals for Optimization In Table 6, we analyze different strategies for retrieving causal hypotheses during scientific discovery. The results show that combining hypotheses associated with both increases and decreases in property fitness from reference to target leads to more effective discovery than exclusively retrieving hypotheses associated with either direction. This suggests that diversifying learning signals enables language models to learn from a broader range of observational pairs and causal hypotheses, thereby supporting the generation of more accurate hypotheses and scientific principles aligned with the discovery objective, validating our design choice explained in Appendix Section A.

Furthermore, retrieving hypotheses with consistently increasing or decreasing gains provides stronger learning signals than random retrieval, where informative signals may be diluted. Notably, PRINCIPIA consistently outperforms the plain optimization algorithm even when using diverse hypothesis retrieval strategies. This suggests that the quality of the hypotheses and principles iteratively generated and refined by PRINCIPIA remains effective across different retrieval strategies, supporting the overall quality of the constructed principles and the effectiveness of PRINCIPIA.

Efficacy in Multi-Objective Discovery While we have demonstrated the effectiveness of PRINCIPIA across various discovery tasks, most involve single-objective optimization. To evaluate its robustness under diverse objectives, we consider a multi-objective protein sequence engineering task that maximizes the Syn-3bfo fitness score, equivalent to the normalized negative Potts-model energy on the PDB 3bfo-derived synthetic landscape, while minimizing the Hamming distance from the wild-type sequence. Following Wang et al. (2025b), we scalarize the two objectives with equal weights. As shown in Figure 7, PRINCIPIA improves over the plain optimization algorithm, while other knowledge-augmented methods degrade over plain optimization algorithm, demonstrating the robustness of test-time causal principle augmentation across diverse discovery objectives.

Analysis with additional metrics To further demonstrate the effectiveness of PRINCIPIA in accelerating scientific discovery through test-time constructed principles, we evaluate its performance using additional metrics. In Table 7, we report the Hit Ratio after excluding essential genes from the ground-truth set, following the additional evaluation protocol adopted in Roohani et al. (2025). PRINCIPIA achieves the highest Hit Ratio on both the Scharenberg22 and IFNG gene perturbation tasks, outperforming the baseline optimization algorithms across both tasks.

Furthermore, Figure 9 presents the Top-100 Average Score trends with respect to the number of oracle calls, complementary to the Top-10 AUC and Top-100 Average scores after a fixed amount of oracle calls. The results show that PRINCIPIA consistently achieves higher Top-100 Average Scores than the existing optimization algorithms throughout the optimization process. These results provide further evidence that PRINCIPIA can effectively improve the performance of scientific discovery by constructing and leveraging task-specific principles at test time.

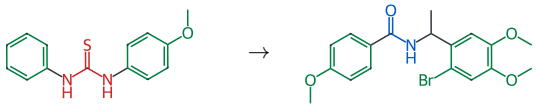
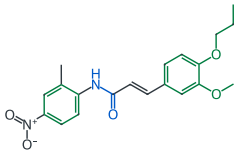
Reference & Target	Original Hypothesis
 Non-inhibitor → Inhibitor	Compared with the reference's **thiourea linker**, the target's **benzamide linker** and **substituted aromatic scaffold** may contribute to CYP2C9 inhibition.
New Observation	**Revised Hypothesis**
 Predicted: Inhibitor **Observed: Non-inhibitor**	An **amide motif** alone does not reliably explain inhibition; its role must be considered in the context of the surrounding **aromatic arrangement**.

Table 8: **Qualifying a hypothesis through new observations.** A comparison between a non-inhibitor and an inhibitor yields hypotheses about CYP2C9 inhibition. These hypotheses were retrieved when predicting a new molecule, whose observed label differed from the prediction. After incorporating new observations, including this case, the revised hypotheses explicitly limit the explanatory role of the amide motif alone. Matching colors link molecular features to their mentions in the text: thiourea (red), amide (blue), and aromatic rings with selected substituents (green). Hypothesis text is summarized; the revision qualifies the existing explanation rather than replacing it entirely.

Qualitative Analysis of Hypothesis Revision Table 8 illustrates how new observations qualify an existing hypothesis in CYP2C9 prediction. The original hypotheses associate the target's inhibitory activity with its benzamide linker and substituted aromatic scaffold. These hypotheses were retrieved when predicting a new amide-containing molecule that was predicted to inhibit CYP2C9 but was observed to be a non-inhibitor. After incorporating new observations, including this case, the revised hypotheses explicitly state that the presence of an amide alone does not reliably explain inhibition, clarifying the applicability of the original explanation.

Robustness of PRINCIPIA to other model variants The results in Figure 8 demonstrate the applicability of PRINCIPIA across different backbone large language models, showing that its effectiveness is not limited to the models used in our main experiments. Specifically, across four backbone models including DeepSeek-V4-Flash (Xu et al., 2026), DeepSeek-V4-Pro (Xu et al., 2026), GPT-5.6-Luna (OpenAI, 2026a), and Gemini-3.1-Flash-Lite (Google DeepMind, 2026), PRINCIPIA achieves higher Top-100 average fitness values than the corresponding baseline methods throughout the optimization process. These results demonstrate that the effectiveness of PRINCIPIA generalizes across different backbone models and scientific discovery tasks.

Optimization Process over Oracle Calls In Figures 8 and 9, we present the trends of the Average Top-100 scores as the number of oracle calls increases. Across 14 different optimization settings, PRINCIPIA outperforms baselines. Notably, the performance advantage becomes more pronounced as additional observations accumulate in tasks such as JNK3, GSK3B, and GB1. These trends suggest that the knowledge constructed during discovery helps guide subsequent evaluations toward more promising candidates, improving the use of the available oracle budget.

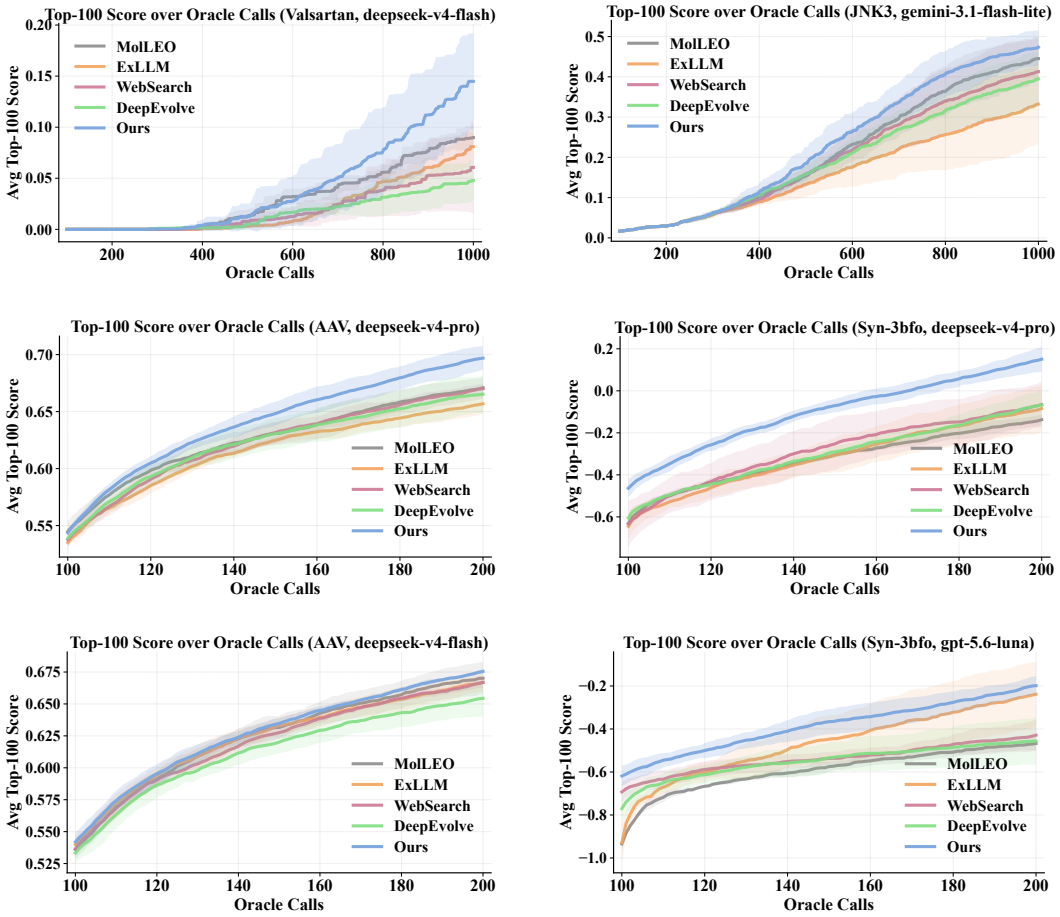


Figure 8: Additional Top-100 average score trends for PRINCIPIA across different backbone models and optimization tasks. The curves are macro-averaged across tasks, with lines representing the mean fitness values over three random seeds and shaded regions indicating the standard deviation.

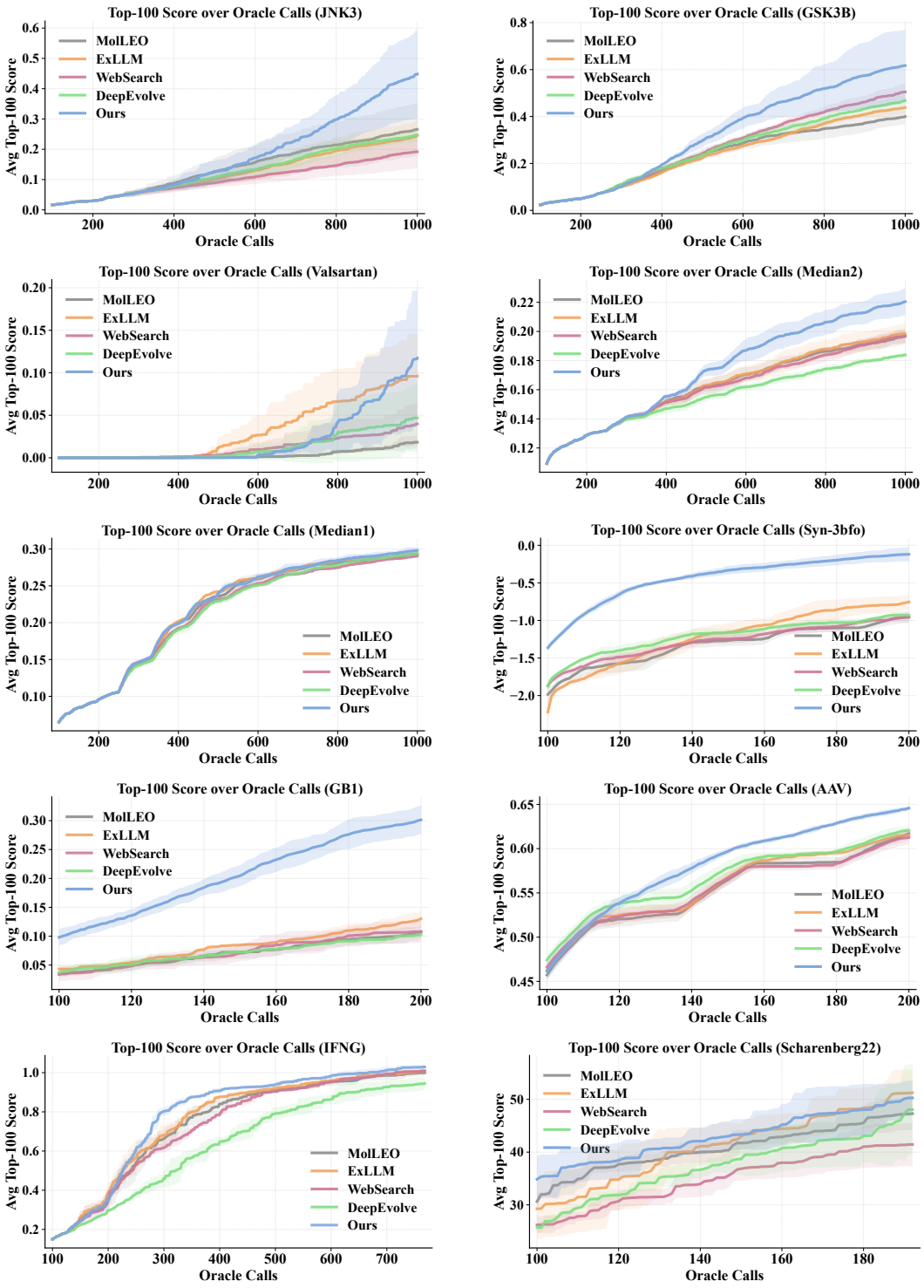


Figure 9: Top-100 average score trends with respect to the number of oracle calls for molecule optimization, protein sequence engineering, and gene perturbation tasks reported in Tables 1 and 2. The curves are macro-averaged across tasks, with lines representing the mean fitness values over three random seeds and shaded regions indicating the standard deviation.

F PROMPTS

In this section, we provide the prompts used for PRINCIPIA in our experiments, along with those used for the baseline methods.

F.1 PROMPT FOR MOLECULE OPTIMIZATION

Figure 10 shows the prompt for comparative causal hypothesis generation, and Figure 11 shows the prompt for principle induction. Figure 12 and Figure 13 provide the hypothesis and principle retrieval prompts, respectively. Figure 14 presents the prompt for hypothesis revision using new observations. Figure 15 shows the candidate generation prompt. Figure 16 lists the task properties, descriptions, and objectives.

F.2 PROMPT FOR PROTEIN ENGINEERING

Figure 17 shows the prompt for comparative causal hypothesis generation, and Figure 18 shows the prompt for principle induction. Figure 19 and Figure 20 provide the hypothesis and principle retrieval prompts, respectively. Figure 21 presents the prompt for hypothesis revision using new observations. Figure 22 shows the candidate generation prompt. Figure 23 lists the task properties, descriptions, and objectives.

F.3 PROMPT FOR GENE PERTURBATION

Figure 24 shows the prompt for comparative causal hypothesis generation, and Figure 25 shows the prompt for principle induction. Figure 26 and Figure 27 provide the hypothesis and principle retrieval prompts, respectively. Figure 28 presents the prompt for hypothesis revision using new observations. Figure 29 shows the candidate generation prompt, including instructions to exclude previously proposed genes. Figure 30 lists the task properties, descriptions, and objectives.

F.4 PROMPT FOR MOLECULAR PROPERTY PREDICTION

Figure 31 shows the prompt for comparative causal hypothesis generation, and Figure 32 shows the prompt for principle induction. Figure 33 and Figure 34 provide the hypothesis and principle retrieval prompts, respectively. Figure 35 presents the prompt for hypothesis revision using new observations. Figure 36 shows the prompt for held-out prediction using retrieved hypotheses and principles. Figure 37 lists the eight task descriptions, questions, observed labels, and answer formats. These templates correspond to the six-round sequential setting. Braced placeholders denote task-specific text or runtime data; block placeholders contain the corresponding comparison records, cluster members, hypotheses, or observations.

Figure 10: Prompt for Comparative Causal Hypothesis Generation on Molecular Optimization

```
You are a computational chemist analyzing the results of molecular
optimization.

The following molecular transformation was performed during an evolutionary
optimization process:

Parent molecule:
{parent_info}

Child molecule:
SMILES: {child_smiles}
Fitness score:  {child_fitness:.4f}

Target property:  {target_property}
Task Description:  {task_description}

Fitness change:  {fitness_change_description}
```

```
Based on the structural differences between the parent and the child
molecule, and the resulting fitness change, provide a concise causal
hypothesis explaining why this transformation led to the observed fitness
change.

Return the output as a valid JSON object with the following structure only:

{
{
"hypotheses":  [
{
{
"causality":  "A free-text mechanism description",
"evidence_features":  ["feature1", ...]
}
},
...]
}
}

Requirements:
- "causality" must be a concise explanation of why the structural
transformation likely caused the observed fitness change.
- Focus on specific structural modifications such as functional group
additions/removals, ring changes, scaffold changes, or atom substitutions.
- "evidence_features" must contain short phrases naming the key structural
changes that support the hypothesis.  - Output only the JSON object and
nothing else.
```

Figure 11: Prompt for Principle Induction on Molecular Optimization

```
You are summarizing a molecular similarity cluster from an evolutionary
optimization run.

You will be given multiple molecules from the same cluster.  For each
molecule, you will see:
- SMILES
- Fitness score ({target_property})
- Hypotheses from parent→child transformations (if any)

Your task is to extract a short, reusable, cluster-specific semantic memory
that captures  structural patterns that are associated with fitness changes.

Internally do this before producing the final JSON:
1.  Identify the structural features that recur across the SMILES in this
cluster.
2.  Separate core scaffold features from peripheral substituent variation.
3.  Check whether the hypotheses support those recurring structural
features.
4.  Remove any statement that is too generic (e.g., "adding functional groups
improves fitness").
5.  Keep only features that are specific to this cluster.

Priority order:
1.  Shared scaffold / chemotype
2.  Recurring substituent or functional group patterns linked to fitness
improvement
3.  Structural modifications that consistently increased or decreased
fitness
4.  Contradictions or inconsistencies across members
Definition of a good cluster memory:
- Names the recurring scaffold shape or motif
- Explains what structural changes improve vs.  harm fitness
- Is specific enough to guide future mutations for similar molecules

Definition of a bad cluster memory:
- Mostly generic chemistry language without cluster-specific detail
```

```
- Ignores the actual recurring scaffold in the SMILES
- Restates hypotheses verbatim without synthesis
- Explicitly lists the index of each molecule as evidence
Return JSON:
{{
"cluster_summary": "...",
"patterns": {{
"positive": ["structural patterns linked to higher fitness..."],
"negative": ["structural patterns linked to lower fitness..."]
}},
"contradictions": ["...", "..."]
}}
Output constraints:
- Mention the actual recurring scaffold in cluster_summary.
- Use cautious frequency qualifiers (e.g., "tends to", "often").
- If all transformations improved fitness, set "contradictions" to [].
- Do not include claims unsupported by the cluster examples.

Target property: {target_property}

Cluster data:
{cluster_examples}
```

Figure 12: Prompt for Hypothesis Retrieval on Molecular Optimization

```
You are selecting the most relevant episodic memories for a molecular
optimization step.

Query molecules:
{query_molecules_text}

Target property: {target_property}
Task Description: {task_description}

You will receive a list of candidate episodic memories. Each candidate
includes:
- an index
- a parent molecule and child molecule
- the observed fitness change
- the inferred hypotheses for that transformation

Select the indices of the memories that are most relevant for guiding the
next mutation or crossover of the query molecules.

Selection priorities:
1. Hypotheses that are structurally relevant to the query molecules
2. Transformations with actionable, reusable guidance
3. Transformations that improved fitness, unless a decreasing-fitness
example is clearly useful as a warning
4. Avoid redundant memories that say essentially the same thing

Return only valid JSON with this schema:
{{
"indices": [0, 3, 5]
}}

Requirements:
- Select at most {top_k} indices.
- Indices must come only from the provided candidate list.
- If fewer than {top_k} are useful, return fewer.
- Output JSON only.

Candidate episodic memories:
{candidate_memories_text}
```

Figure 13: Prompt for Principle Retrieval on Molecular Optimization

```
You are selecting the most relevant semantic memories for a molecular
optimization step.

Query molecules:
{query_molecules_text}

Target property: {target_property}
Task Description: {task_description}

You will receive a list of candidate semantic memories. Each candidate
is a cluster-level summary of recurring structural patterns observed during
optimization.

Select the indices of the semantic memories that are most relevant for
guiding the next mutation or crossover of the query molecules.

Selection priorities:
1. Cluster summaries and patterns that best match the likely scaffold or
substituent space of the query molecules
2. Memories with concrete positive/negative structural guidance
3. Diverse but complementary guidance when selecting multiple memories
4. Avoid redundant clusters with overlapping advice

Return only valid JSON with this schema:
{{
"indices": [1, 2]
}}

Requirements:
- Select at most {top_k} indices.
- Indices must come only from the provided candidate list.
- If fewer than {top_k} are useful, return fewer.
- Output JSON only.

Candidate semantic memories:
{candidate_memories_text}
```

Figure 14: Prompt for Hypothesis Revision on Molecular Optimization

```
You are a computational chemist refining causal hypotheses about molecular
optimization.

During an evolutionary optimization process, a source molecule was mutated
into a target molecule.
The hypotheses below were originally inferred to explain this transformation.
Now, additional mutation outcomes (observations) have been collected from
other transformations.
Use these observations to refine the hypotheses.

=== Edge Being Refined ===
Source molecule:
SMILES: {source_smiles}
Fitness score: {source_fitness:.4f}

Target molecule:
SMILES: {target_smiles}
Fitness score: {target_fitness:.4f}

Target property: {target_property}
Task Description: {task_description}

=== Current Hypotheses for This Edge ===
{current_hypotheses_text}

=== Observations from Other Mutations ===
```

```
{observations_text}

Based on the observations above, please refine the current hypotheses for
the source→target edge.  Consider the following:
- Which hypotheses are supported by the observed outcomes?  Keep and
strengthen them.
- Which hypotheses are contradicted by the outcomes?  Revise or remove them.
- Are there any new patterns in the observations that suggest new hypotheses?
Add them.

Return the output as a valid JSON object with the following structure only:

{{
"hypotheses":  [
{{
"causality":  "A free-text mechanism description",
"evidence_features":  ["feature1", ...]}},
...]}}

Requirements:  - "causality" must be a concise explanation of the refined
causal mechanism.
- "evidence_features" must contain short phrases naming the key structural
changes that support the hypothesis.
- Output only the JSON object and nothing else.
```

Figure 15: Prompt for Candidate Generation on Molecular Optimization

```
You are a molecular designer tasked with creating chemical analogs by
mutating parent smiles strings.

Given the following molecule:
Parent SMILES: {parent_smiles}
Parent {target_property} score:  {parent_score}
Target property:  {target_property}
Task Description:  {task_description}
Task Objective:  {task_objective}

{in_task_experience_memory}
Requirements:
1.  Each analog should be chemically valid
2.  Propose exactly {num_mutations} mutated molecules

Output format:
Return only the SMILES strings, one per line in /box{{SMILES_HERE}}, without
any additional text or explanations.  YOU MUST FOLLOW THIS FORMAT.
```

Figure 16: Task Property, Description, and Objective on Molecular Optimization

```
Task Property:  JNK3
Task Description:  I have a molecule and their JNK3 scores.  The JNK3 score
measures a moleculars biological activity against JNK3.
Task Objective:  Please propose a new molecule that has a higher JNK3 score.
You can either make mutations based on the given molecules or just propose a
new molecule based on your knowledge.

Task Property:  GSK3β
Task Description:  I have a molecule and their GSK3β scores.  The GSK3β
score measures a moleculars biological activity against Glycogen Synthase
Kinase 3 Beta.
Task Objective:  Please propose a new molecule that has a higher GSK3β score.
You can either make mutations based on the given molecules or just propose a
new molecule based on your knowledge.

Task Property:  Median1
Task Description:  I have a molecule and their median1 scores.  The median1
```

```
score measures the geometric mean of a moleculeś ECFC4-fingerprint Tanimoto
similarities to both camphor and menthol.
Task Objective:  Please propose a new molecule that has a higher median1
score.  You can either make mutations based on the given molecules or just
propose a new molecule based on your knowledge.

Task Property:  Median2
Task Description:  I have a molecule and their median2 scores.  The median2
score measures the geometric mean of a moleculeś ECFC6-fingerprint Tanimoto
similarities to both tadalafil and sildenafil.
Task Objective:  Please propose a new molecule that has a higher median2
score.  You can either make mutations based on the given molecules or just
propose a new molecule based on your knowledge.

Task Property:  Valsartan Smarts
Task Description:  I have a molecule and their valsartan SMARTS scores.
The valsartan SMARTS score measures whether a molecule contains a
valsartan-related pharmacophore SMARTS pattern while matching target
physicochemical properties (LogP, TPSA and Bertz index).
Task Objective:  Please propose a new molecule that has a higher valsartan
SMARTS score.  You can either make mutations based on the given molecules or
just propose a new molecule based on your knowledge.
```

Figure 17: Prompt for Comparative Causal Hypothesis Generation on Protein Engineering

```
You are a computational biologist analyzing the results of protein sequence
optimization.

The following sequence mutation was performed during an evolutionary
optimization process:

Parent sequence:
{parent_info}

Child sequence:
Sequence:  {child_sequence}
Fitness score:  {child_fitness:.4f}

Target property:  {target_property}
Task Description:  {task_description}

Fitness change:  {fitness_change_description}

Based on the differences between the parent and the child sequence, and
the resulting fitness change, provide a concise causal hypothesis explaining
why this mutation led to the observed fitness change.

Return the output as a valid JSON object with the following structure only:

{{
"hypotheses":  [{{
"causality":  "A free-text mechanism description",
"evidence_features":  ["feature1", ...]}},
...]}}

Requirements:
- "causality" must be a concise explanation of why the sequence change likely
caused the observed fitness change.
- Focus on specific residue substitutions, changes at conserved positions, or
motif changes.
- "evidence_features" must contain short phrases naming the key sequence
changes that support the hypothesis.
- Output only the JSON object and nothing else.
```

Figure 18: Prompt for Principle Induction on Protein Engineering

```
You are summarizing a protein sequence similarity cluster from an
evolutionary optimization run.

You will be given multiple protein sequences from the same cluster.  For
each sequence, you will see:
- Sequence
- Fitness score ({target_property})
- Hypotheses from parent→child mutations (if any)

Your task is to extract a short, reusable, cluster-specific semantic memory
that captures  sequence patterns that are associated with fitness changes.

Internally do this before producing the final JSON:
1.  Identify the residues/motifs that recur across the sequences in this
cluster.
2.  Separate conserved core positions from variable positions.
3.  Check whether the hypotheses support those recurring sequence features.
4.  Remove any statement that is too generic (e.g., "mutations improve
fitness").
5.  Keep only features that are specific to this cluster.
Priority order:
1.  Shared conserved motif / consensus positions
2.  Recurring residue patterns at specific positions linked to fitness
improvement
3.  Residue substitutions that consistently increased or decreased fitness
4.  Contradictions or inconsistencies across members
Definition of a good cluster memory:
- Names the recurring sequence motif or conserved positions
- Explains what residue changes improve vs.  harm fitness
- Is specific enough to guide future mutations for similar sequences

Definition of a bad cluster memory:
- Mostly generic language without cluster-specific detail
- Ignores the actual recurring motif in the sequences
- Restates hypotheses verbatim without synthesis
- Explicitly lists the index of each sequence as evidence
Return JSON:
{{
"cluster_summary":  "...",
"patterns":  {{
"positive":  ["sequence patterns linked to higher fitness..."],
"negative":  ["sequence patterns linked to lower fitness..."]
}},
"contradictions":  ["...", "..."]
}}

Output constraints:
- Mention the actual recurring motif or conserved positions in
cluster_summary.
- Use cautious frequency qualifiers (e.g., "tends to", "often").
- If all mutations improved fitness, set "contradictions" to [].
- Do not include claims unsupported by the cluster examples.

Target property:  {target_property}

Cluster data:
{cluster_examples}
```

Figure 19: Prompt for Hypothesis Retrieval on Protein Engineering

```
You are selecting the most relevant episodic memories for a protein sequence
optimization step.

Query sequences:
{query_sequences_text}
```

```
Target property: {target_property}
Task Description: {task_description}

You will receive a list of candidate episodic memories. Each candidate
includes:
- an index
- a parent sequence and child sequence
- the observed fitness change
- the inferred hypotheses for that mutation
Select the indices of the memories that are most relevant for guiding the
next mutation or crossover of the query sequences.

Selection priorities:
1. Hypotheses that are relevant to the query sequences
2. Mutations with actionable, reusable guidance
3. Mutations that improved fitness, unless a decreasing-fitness example is
clearly useful as a warning
4. Avoid redundant memories that say essentially the same thing

Return only valid JSON with this schema:
{{
"indices": [0, 3, 5]
}}

Requirements:
- Select at most {top_k} indices.
- Indices must come only from the provided candidate list.
- If fewer than {top_k} are useful, return fewer.
- Output JSON only.

Candidate episodic memories:
{candidate_memories_text}
```

Figure 20: Prompt for Principle Retrieval on Protein Engineering

```
You are selecting the most relevant semantic memories for a protein sequence
optimization step.

Query sequences:
{query_sequences_text}

Target property: {target_property}
Task Description: {task_description}

You will receive a list of candidate semantic memories. Each candidate
is a cluster-level summary of recurring sequence patterns observed during
optimization.

Select the indices of the semantic memories that are most relevant for
guiding the next mutation or crossover of the query sequences.

Selection priorities:
1. Cluster summaries and patterns that best match the likely sequence/motif
space of the query sequences
2. Memories with concrete positive/negative sequence-level guidance
3. Diverse but complementary guidance when selecting multiple memories
4. Avoid redundant clusters with overlapping advice

Return only valid JSON with this schema:
{{"indices": [1, 2]}}

Requirements:
- Select at most {top_k} indices.
- Indices must come only from the provided candidate list.
- If fewer than {top_k} are useful, return fewer.
- Output JSON only.
```

```
Candidate semantic memories:
{candidate_memories_text}
```

Figure 21: Prompt for Hypothesis Revision on Protein Engineering

```
You are a computational biologist refining causal hypotheses about protein
sequence optimization.

During an evolutionary optimization process, a source sequence was mutated
into a target sequence.
The hypotheses below were originally inferred to explain this mutation.
Now, additional mutation outcomes (observations) have been collected from
other mutations.
Use these observations to refine the hypotheses.

=== Edge Being Refined ===
Source sequence:
Sequence: {source_sequence}
Fitness score: {source_fitness:.4f}

Target sequence:
Sequence: {target_sequence} Fitness score: {target_fitness:.4f}
Target property: {target_property}
Task Description: {task_description}

=== Current Hypotheses for This Edge ===
{current_hypotheses_text}

=== Observations from Other Mutations ===
{observations_text}

Based on the observations above, please refine the current hypotheses for
the source→target edge.
Consider the following:
- Which hypotheses are supported by the observed outcomes? Keep and
strengthen them.
- Which hypotheses are contradicted by the outcomes? Revise or remove them.
- Are there any new patterns in the observations that suggest new hypotheses?
Add them.

Return the output as a valid JSON object with the following structure only:

{{
"hypotheses": [
{{
"causality": "A free-text mechanism description",
"evidence_features": ["feature1", ...]
}}
, ...
]
}}

Requirements:
- "causality" must be a concise explanation of the refined causal mechanism.
- "evidence_features" must contain short phrases naming the key sequence
changes that support the hypothesis.
- Output only the JSON object and nothing else.
```

Figure 22: Prompt for Candidate Generation on Protein Engineering

```
You are a protein engineer tasked with creating variants by mutating parent
amino-acid sequences.

Given the following protein sequence:
Parent sequence: {parent_sequence}
```

```
Parent {target_property} score: {parent_score}

Target property: {target_property}
Task Description: {task_description}
Task Objective: {task_objective}
{in_task_experience_memory}

Requirements:
1. Each variant must be a valid sequence over the 20 standard amino acids
with the SAME length as the parent
2. Propose exactly {num_mutations} mutated sequences
Output format:
Return only the amino-acid sequences, one per line in /box{{SEQUENCE_HERE}},
without any additional text or explanations. YOU MUST FOLLOW THIS FORMAT.
```

Figure 23: Task Property, Description, and Objective on Protein Engineering

```
Task Property: Syn-3bfo Potts Fitness
Task Description: I have protein sequences and their Syn-3bfo fitness
scores. The score is a Potts-model fitness (the normalized negative energy)
over a synthetic fitness landscape derived from PDB 3bfo, and a higher score
is better.
Task Objective: Please propose a new protein sequence that has a higher
Syn-3bfo fitness score. You can either mutate the given sequences or
propose a new sequence based on your knowledge. Keep the same sequence
length and use only the 20 standard amino acids.

Task Property: GB1 Fitness
Task Description: I have protein sequences and their GB1 fitness scores.
GB1 is the B1 domain of protein G; the fitness score measures the
binding/stability of a short variable region, and a higher score is better.
Task Objective: Please propose a new protein sequence that has a higher GB1
fitness score. You can either mutate the given sequences or propose a new
sequence based on your knowledge. Keep the same sequence length and use only
the 20 standard amino acids.

Task Property: AAV Capsid Fitness
Task Description: I have protein sequences and their AAV fitness scores.
The score is predicted by a machine-learning model of AAV2 capsid protein
viability over a short variable region, and a higher score is better.
Task Objective: Please propose a new protein sequence that has a higher AAV
fitness score. You can either mutate the given sequences or propose a new
sequence based on your knowledge. Keep the same sequence length and use only
the 20 standard amino acids.

Task Property: Syn-3bfo Potts fitness (Weighted Objective)
Task Description: I have protein sequences and their Syn-3bfo fitness
scores. The score is a Potts-model fitness (the normalized negative energy)
over a synthetic fitness landscape derived from PDB 3bfo, and a higher score
is better. This is a multi-objective task: the score is a weighted sum
(equal weights) of the raw fitness and a closeness term, 1 - (normalized
Hamming distance) to a fixed reference sequence. A higher score means higher
fitness while staying close to the reference (fewer edits from it).
Reference sequence: {reference_sequence}
Task Objective: Please propose a new protein sequence that has a higher
Syn-3bfo fitness score. You can either mutate the given sequences or
propose a new sequence based on your knowledge. Keep the same sequence
length and use only the 20 standard amino acids.
```

Figure 24: Prompt for Comparative Causal Hypothesis Generation on Gene Perturbation

```
You are a computational biologist analyzing the results of a
gene-perturbation screen.

The following gene proposal was made during an evolutionary discovery
```

```
process:

Parent gene(s):
{parent_info}

Proposed gene:
Gene: {child_gene}
Score: {child_fitness:.4f}

Research goal: {target_property}
Task Description: {task_description}

Score change: {fitness_change_description}

Based on the biological relationship between the parent gene(s) and the
proposed gene, and the resulting score change, provide a concise causal
hypothesis explaining why proposing this gene led to the observed score
change.

Return the output as a valid JSON object with the following structure only:

{{
"hypotheses": [
{{
"causality": "A free-text mechanism
description",
"evidence_features": ["feature1", ...]
}},...]}}

Requirements:
- "causality" must be a concise explanation of why proposing this gene likely
caused the observed score change.
- Focus on biological relationships such as shared pathways, complexes,
regulators, effectors, or interaction partners.
- "evidence_features" must contain short phrases naming the key biological
relationships that support the hypothesis.
- Output only the JSON object and nothing else.
```

Figure 25: Prompt for Principle Induction on Gene Perturbation

```
You are summarizing a gene cluster from an evolutionary gene-perturbation
discovery run.

You will be given multiple genes from the same cluster. For each gene, you
will see:
- Gene symbol (HGNC)
- Fitness score (target_property)
- Hypotheses from parent→child transformations (if any)

Your task is to extract a short, reusable, cluster-specific semantic memory
that captures functional/biological patterns that are associated with
fitness changes.
Internally do this before producing the final JSON:
1. Identify the biological themes that recur across the genes in this
cluster (shared pathway, complex, functional family, or regulatory role).
2. Separate core shared function from peripheral gene-specific variation.
3. Check whether the hypotheses support those recurring functional themes.
4. Remove any statement that is too generic (e.g., "perturbing important
genes improves the score").
5. Keep only patterns that are specific to this cluster.

Priority order:
1. Shared pathway / protein complex / functional family
2. Recurring regulatory or mechanistic roles linked to fitness improvement
3. Perturbations that consistently increased or decreased fitness
4. Contradictions or inconsistencies across members
```

```
Definition of a good cluster memory:
- Names the recurring pathway, complex, or functional theme
- Explains what kinds of genes improve vs. harm fitness
- Is specific enough to guide future gene proposals for similar genes

Definition of a bad cluster memory:
- Mostly generic biology language without cluster-specific detail
- Ignores the actual recurring function shared by the genes
- Restates hypotheses verbatim without synthesis
- Explicitly lists the index of each gene as evidence
Return JSON:
{{
"cluster_summary": "...",
"patterns": {{
"positive": ["functional patterns linked to higher fitness..."],
"negative": ["functional patterns linked to lower fitness..."]
}},
"contradictions": ["...", "..."]
}}

Output constraints:
- Mention the actual recurring pathway/function in cluster_summary.
- Use cautious frequency qualifiers (e.g., "tends to", "often").
- If all transformations improved fitness, set "contradictions" to [].
- Do not include claims unsupported by the cluster examples.
Target property: {target_property}
Cluster data:
{cluster_examples}
```

Figure 26: Prompt for Hypothesis Retrieval on Gene Perturbation

```
You are selecting the most relevant episodic memories for a gene-perturbation
discovery step.

Query genes:
{query_genes_text}

Research goal: {target_property}
Task Description:{task_description}

You will receive a list of candidate episodic memories. Each candidate
includes:
- an index
- a parent gene and proposed gene
- the observed score change
- the inferred hypotheses for that proposal

Select the indices of the memories that are most relevant for guiding the
next gene proposal for the query genes.

Selection priorities:
1. Hypotheses that are biologically relevant to the query genes
2. Proposals with actionable, reusable guidance
3. Proposals that improved the score, unless a decreasing-score example is
clearly useful as a warning
4. Avoid redundant memories that say essentially the same thing

Return only valid JSON with this schema:
{{
"indices": [0, 3, 5]
}}

Requirements:
- Select at most {top_k} indices.
- Indices must come only from the provided candidate list.
- If fewer than {top_k} are useful, return fewer.
- Output JSON only.
```

```
Candidate episodic memories:
{candidate_memories_text}
```

Figure 27: Prompt for Principle Retrieval on Gene Perturbation

```
You are selecting the most relevant semantic memories for a gene-perturbation
discovery step.

Query genes:
{query_genes_text}

Research goal: {target_property}
Task Description: {task_description}

You will receive a list of candidate semantic memories. Each candidate is
a cluster-level summary of recurring gene/pathway patterns observed during
discovery.

Select the indices of the semantic memories that are most relevant for
guiding the next gene proposal for the query genes.

Selection priorities:
1. Cluster summaries and patterns that best match the likely pathways or
functions of the query genes
2. Memories with concrete positive/negative biological guidance
3. Diverse but complementary guidance when selecting multiple memories
4. Avoid redundant clusters with overlapping advice

Return only valid JSON with this schema:
{{
"indices": [1, 2]
}}

Requirements:
- Select at most {top_k} indices.
- Indices must come only from the provided candidate list.
- If fewer than {top_k} are useful, return fewer.
- Output JSON only.

Candidate semantic memories:
{candidate_memories_text}
```

Figure 28: Prompt for Hypothesis Revision on Gene Perturbation

```
You are a computational biologist refining causal hypotheses about
gene-perturbation discovery.

During an evolutionary discovery process, a source gene led to proposing
a target gene.
The hypotheses below were originally inferred to explain this relationship.
Now, additional proposal outcomes (observations) have been collected from
other proposals.
Use these observations to refine the hypotheses.

=== Edge Being Refined ===
Source gene:
Gene: {source_gene}
Score: {source_fitness:.4f}

Target gene:
Gene: {target_gene} Score: {target_fitness:.4f}
Research goal: {target_property} Task Description: {task_description}
=== Current Hypotheses for This Edge ===
{current_hypotheses_text}
```

```
=== Observations from Other Proposals ===
{observations_text}
Based on the observations above, please refine the current hypotheses for
the source->target edge.  Consider the following:
- Which hypotheses are supported by the observed outcomes?  Keep and
strengthen them.
- Which hypotheses are contradicted by the outcomes?  Revise or remove them.
- Are there any new patterns in the observations that suggest new hypotheses?
Add them.

Return the output as a valid JSON object with the following structure only:

{{
"hypotheses":  [
{{
"causality":  "A free-text mechanism
description",
"evidence_features":  ["feature1", ...]
}},
...]}}

Requirements:
- "causality" must be a concise explanation of the refined causal mechanism.
- "evidence_features" must contain short phrases naming the key biological
relationships that support the hypothesis.
- Output only the JSON object and nothing else.
```

Figure 29: Prompt for Candidate Generation on Gene Perturbation

```
You are a geneticist proposing new candidate genes to perturb, informed by
two parent genes and your biological knowledge.
Given the following two parent genes and their scores:

Parent 1 gene:  {parent_gene_1}
Parent 1 score:  {parent_score_1}
Parent 2 gene:  {parent_gene_2}
Parent 2 score:  {parent_score_2}
Research goal:  {target_property}

Task Description:  {task_description}
Task Objective:  {task_objective}
{in_task_experience_memory}

You may propose genes related to either or both parents (shared pathways,
complexes, regulators, effectors, or interaction partners), or propose
entirely new genes informed by the parents and your knowledge of the
relevant biology.  Prioritize proposals likely to improve upon the parents'
current scores.
Requirements:
1.  Each gene must be a real human gene using its official HGNC symbol
2.  Prefer novel candidate genes that are unlikely to have been proposed
before, while remaining biologically valid and plausibly relevant to the
research goal
3.  Propose exactly {num_children} new candidate genes
Output format:
Return only the gene symbols, one per line in /box{{GENE_HERE}}, without any
additional text or explanations.  YOU MUST FOLLOW THIS FORMAT.

STRICT EXCLUSION
-- genes already proposed in earlier rounds:
The following genes have ALREADY been proposed/tested in previous rounds of
this screen.  You MUST NOT propose any of them again.  Every gene you output
must be different from all of the genes listed below.

{excluded_genes}
```

Figure 30: Task Property, Description, and Objective on Gene Perturbation

```
Task Property: identify genes mediating lysosomal choline recycling using an
endolysosome-focused CRISPR-Cas9 screen
Task Description: I am running a genome-wide CRISPR screen to identify
genes mediating lysosomal choline recycling using an endolysosome-focused
CRISPR-Cas9 screen. For each gene I perturb, I can measure enrichment or
depletion of targeting sgRNAs (indicated by a high score) in the culture
medium lacking free choline, which will be referred to as the score; a higher
score indicates a stronger hit.
Task Objective: Please propose new genes to perturb that are likely to
maximize the score. Use official HGNC gene symbols. You can build on
the genes and scores observed so far or propose new genes based on your
knowledge of the relevant biology and pathways.

Task Property: identify genes that regulate the production of
Interferon-gamma (IFNG)
Task Description: I am running a genome-wide CRISPR screen to identify
genes that regulate the production of Interferon-gamma (IFNG). For each
gene I perturb, I can measure the log fold change in Interferon-gamma
(IFNG) normalized read counts. The score reflects the magnitude of this
effect, so a gene that strongly shifts the phenotype in EITHER direction (a
strong increase or a strong decrease) receives a high score; a higher score
indicates a stronger hit.
Task Objective: Please propose new genes to perturb that are likely to
maximize the score, i.e. genes that produce the largest change in the
phenotype regardless of direction (a strong increase OR a strong decrease).
Use official HGNC gene symbols. You can build on the genes and scores
observed so far or propose new genes based on your knowledge of the relevant
biology and pathways.
```

Figure 31: Prompt for Comparative Causal Hypothesis Generation on Molecular Property Prediction

```
You are a medicinal chemistry expert. Given a target molecule (SMILES)
with its observed label and ONE structurally similar reference molecule,
generate plausible mechanistic hypotheses explaining why the target shows
the observed activity, grounded in how it structurally differs from that
reference.

CRITICAL RULES:
1. Do NOT simply restate the label or predict the label
2. Every hypothesis must be grounded in a concrete structural difference
between the target and the reference
3. If the reference carries the SAME label, the shared substructure is the
likely driver -- name it
4. If the reference carries a DIFFERENT label, the substructure that differs
is the likely driver -- name it
5. Generate 1-2 distinct hypotheses from this single comparison
6. Provide the relevant substructures as evidence features

Task: {task_description}

Target molecule:
SMILES: {smiles}
Observed Label: {label}

Reference molecule:
SMILES: {reference_smiles}
Observed Label: {reference_label} (label {label_agreement})
Tanimoto similarity to target: {similarity}

Output your response as a JSON object with the following format:
{
"hypotheses": [
{
"causality": "A free-text mechanism description",
"evidence_features": ["feature1", ...]
```

```
},
...
]
}
```

Figure 32: Prompt for Principle Induction on Molecular Property Prediction

```
You are summarizing a molecular similarity cluster.

You will be given multiple molecules from the same cluster.
For each molecule, you will see:
- SMILES
- label
- the comparisons it was hypothesised from:  a structurally similar reference
molecule, that reference's label, whether the label CHANGED or stayed
UNCHANGED between the two, and the hypothesis explaining the difference

Your task is to extract a short, reusable, cluster-specific semantic memory.

Internally do this before producing the final JSON:
1.  Identify the structural features that recur across the SMILES.
2.  Separate core scaffold features from peripheral substituent variation.
3.  Check whether the hypotheses support those recurring structural
features.
4.  Contrast the CHANGED comparisons against the UNCHANGED ones:  the
substructures that
recur in CHANGED comparisons are label-determining, the ones that recur in
UNCHANGED
comparisons are tolerated variation.
5.  Remove any statement that is too generic and would fit many unrelated
molecules for the same endpoint.
6.  Keep only features that are local to this cluster.

Priority order
1.  Shared scaffold / chemotype
2.  Substructures that flip the label across the CHANGED comparisons
3.  Substructures that the UNCHANGED comparisons show to be tolerated
4.  Recurring substituent classes
5.  Label-specific interpretation
6.  Generic mechanism language only if still necessary

Definition of a good cluster memory
- Names the recurring scaffold shape or motif
- Explains what is conserved versus variable
- Is specific enough to distinguish this cluster from other positive clusters
for the same task

Definition of a bad cluster memory
- Mostly generic pharmacophore language
- Mostly endpoint-mechanism language
- Statements that ignore the actual recurring scaffold in the SMILES

Return JSON with the following format:
{
"cluster_summary":  "...",
"patterns":  {
"<label_1>":  ["...", "..."],
"<label_2>":  ["...", "..."]
},
"contradictions":  ["...", "..."]
}

Output constraints
- Mention the actual recurring scaffold in cluster_summary.
- Use cautious frequency qualifiers.
- Mention variable regions if present.
- Do not include claims unsupported by the cluster examples.
```

```
- Do not write generic statements unless they are made cluster-specific.
- If all molecules in the cluster share the same label (i.e., there is label
consensus), set "contradictions" to an empty list [].

Cluster data:
{cluster_examples}
```

Figure 33: Prompt for Hypothesis Retrieval on Molecular Property Prediction

```
You are selecting which past molecular comparisons are most useful for
predicting a query molecule's label.

Task: {task_description}

Query molecule: {query_smiles}

Each candidate below is a comparison between a target training molecule
and a reference training molecule, both with known labels, together with the
hypotheses that explain their structural difference.

{candidate_block}

Select the 3 candidates whose structural difference is most informative
for the query molecule.

Selection criteria:
- Prefer comparisons whose differing substructure is actually present in, or
absent from, the query molecule.
- Keep a mix: comparisons where the label changes show what flips the
outcome, comparisons where it stays the same show what is tolerated.
- Prefer specific, mechanistically grounded hypotheses over generic ones.

Return JSON only:
{
"selected_ids": [<candidate id>, ...]
}
```

Figure 34: Prompt for Principle Retrieval on Molecular Property Prediction

```
You are selecting the most relevant semantic memories for a molecular
property prediction.

Task: {task_description}

Query molecule: {query_smiles}

You will receive a list of candidate semantic memories. Each candidate
is a cluster-level
summary of structural patterns recurring among structurally similar training
molecules.

Select the semantic memories that are relevant for predicting the query
molecule's label.

Selection priorities:
1. Clusters whose patterns name a reactive or alerting functional group that
is present in the query molecule, even when the overall scaffold differs
2. Memories with concrete label-discriminative structural guidance
3. Diverse but complementary guidance when selecting multiple memories
4. Avoid redundant clusters with overlapping advice

Return only valid JSON with this schema:
{
"indices": [0, 1]
}
```

```
Requirements:
- Indices must come only from the provided candidate list.
- Output JSON only.

Candidate semantic memories:
{candidate_memories_text}
```

Figure 35: Prompt for Hypothesis Revision on Molecular Property Prediction

```
You are refining causal hypotheses about a molecular structure comparison.

During evaluation, this comparison was retrieved as evidence for predicting
other molecules' labels.
The hypotheses below were originally inferred to explain the comparison.
Now, additional prediction outcomes (observations) have been collected from
the same evaluation round.
Use these observations to refine the hypotheses.

Task: {task_description}

=== Comparison Being Refined ===
Target molecule:
SMILES: {entry_smiles}
Observed label: {entry_label}

Reference molecule:
SMILES: {reference_smiles}
Observed label: {reference_label}

Label Change: {label_change}

=== Current Hypotheses for This Comparison ===
{current_hypotheses_text}

=== Observations from Other Predictions ===
{observations_text}

Based on the observations above, please refine the current hypotheses for
the target-reference comparison.
Consider the following:
- Which hypotheses are supported by the observed outcomes? Keep and
strengthen them.
- Which hypotheses are contradicted by the outcomes? Revise or remove them.
- Are there any new patterns in the observations that suggest new hypotheses?
Add them.

Return the output as a valid JSON object with the following structure only:

{
"hypotheses": [
{
"causality": "A free-text mechanism description",
"evidence_features": ["feature1", ...]
},
...
]
}

Requirements:
- "causality" must be a concise explanation of the refined causal mechanism.
- "evidence_features" must contain short phrases naming the key structural
differences that support the hypothesis.
- Output only the JSON object and nothing else.
```

Figure 36: Prompt for Molecular Property Prediction

```
{task_description}

=== Retrieved Knowledge ===

## Semantic Knowledge:
{retrieved_principles}

## Structural Comparisons and Their Hypotheses:
{retrieved_comparisons}

=== End Retrieved Knowledge ===

{task_question}
Molecule: {query_smiles}

Answer format: {answer_format}
```

Figure 37: Task Property, Description, and Objective on Molecular Property Prediction

```
Task Property: DILI
Task Description:
You are a drug discovery assistant tasked with predicting the DILI
(Drug-Induced Liver Injury) of a molecule. Specifically, your role is to
determine whether a molecule is likely to cause liver injury or not.
Task Objective / Question:
Determine whether the given molecule can cause liver injury or not.
Observed Labels: Can cause liver injury; Does not cause liver injury
Answer Format:
Your final answer should be formatted as either: 'Final answer: Can cause
liver injury' or 'Final answer: Does not cause liver injury.'

Task Property: hERG
Task Description:
You are a drug discovery assistant tasked with predicting the hERG (Human
ether-a-go-go related gene (hERG)) inhibition of a molecule. Specifically,
your role is to determine whether a molecule blocks or does not block the
hERG.
Task Objective / Question:
Determine the hERG inhibition of the given molecule.
Observed Labels: Block; Not block
Answer Format:
Your final answer should be formatted as either: 'Final answer: Block' or
'Final answer: Not block'

Task Property: Ames
Task Description:
You are a drug discovery assistant tasked with predicting the Ames
mutagenicity of a molecule. Specifically, your role is to determine whether
a molecule is mutagenic or not mutagenic.
Task Objective / Question:
Determine whether the given molecule is mutagenic or not mutagenic.
Observed Labels: Mutagenic; Not mutagenic
Answer Format:
Your final answer should be formatted as either: 'Final answer: Mutagenic'
or 'Final answer: Not mutagenic.'

Task Property: Skin Reaction
Task Description:
You are a drug discovery assistant tasked with predicting the skin reaction
of a molecule. Specifically, your role is to determine whether a molecule
induces an immune reaction in inherently susceptible individuals that leads
to skin sensitization.
Task Objective / Question:
Determine whether the given molecule induces an immune reaction in inherently
susceptible individuals that leads to skin sensitization.
```

```
Observed Labels:  Cause skin reaction; Does not cause skin reaction
Answer Format:
Your final answer should be formatted as either:  'Final answer:  Cause skin
reaction' or 'Final answer:  Does not cause skin reaction.'

Task Property:  CYP1A2
Task Description:
You are a drug discovery assistant tasked with predicting the inhibition of
the cytochrome P450 1A2 (CYP1A2) enzyme by a molecule.  Specifically, your
role is to determine whether a molecule inhibits CYP1A2 or does not inhibit
CYP1A2.
Task Objective / Question:
Determine whether the given molecule inhibits CYP1A2.
Observed Labels:  Does not inhibit CYP1A2; Inhibits CYP1A2
Answer Format:
Your final answer should be formatted as either:  'Final answer:  Inhibits
CYP1A2' or 'Final answer:  Does not inhibit CYP1A2.'

Task Property:  CYP2C9
Task Description:
You are a drug discovery assistant tasked with predicting the inhibition of
the cytochrome P450 2C9 (CYP2C9) enzyme by a molecule.  Specifically, your
role is to determine whether a molecule inhibits CYP2C9 or does not inhibit
CYP2C9.
Task Objective / Question:
Determine whether the given molecule inhibits CYP2C9.
Observed Labels:  Does not inhibit CYP2C9; Inhibits CYP2C9
Answer Format:
Your final answer should be formatted as either:  'Final answer:  Inhibits
CYP2C9' or 'Final answer:  Does not inhibit CYP2C9.'

Task Property:  CYP2D6
Task Description:
You are a drug discovery assistant tasked with predicting the inhibition of
the cytochrome P450 2D6 (CYP2D6) enzyme by a molecule.  Specifically, your
role is to determine whether a molecule inhibits CYP2D6 or does not inhibit
CYP2D6.
Task Objective / Question:
Determine whether the given molecule inhibits CYP2D6.
Observed Labels:  Does not inhibit CYP2D6; Inhibits CYP2D6
Answer Format:
Your final answer should be formatted as either:  'Final answer:  Inhibits
CYP2D6' or 'Final answer:  Does not inhibit CYP2D6.'

Task Property:  CYP3A4
Task Description:
You are a drug discovery assistant tasked with predicting the inhibition of
the cytochrome P450 3A4 (CYP3A4) enzyme by a molecule.  Specifically, your
role is to determine whether a molecule inhibits CYP3A4 or does not inhibit
CYP3A4.
Task Objective / Question:
Determine whether the given molecule inhibits CYP3A4.
Observed Labels:  Does not inhibit CYP3A4; Inhibits CYP3A4
Answer Format:
Your final answer should be formatted as either:  'Final answer:  Inhibits
CYP3A4' or 'Final answer:  Does not inhibit CYP3A4.'
```