When Do LLMs Improve Bayesian Optimization? A Systematic Comparison Across Molecular and Protein Design

Anonymous Author(s)
Affiliation
Address
email

Abstract

While powerful, classical Bayesian Optimization (BO) and active learning methods struggle to incorporate complex prior knowledge, provide limited interpretability in explaining why a candidate looks promising, and can be computationally demanding. Large language models (LLMs) offer complementary strengths in reasoning ability and integration of domain knowledge, but it remains unclear **when** and **how** they can reliably improve BO campaigns. We reconcile previous reports by providing a systematic comparison of various LLM-based approaches (off-the-shelf reasoning LLMs relying on in-context learning, fine-tuned on synthetic BO, and light-weight agentic workflows using tools) against classical statistical BO across molecular optimization and protein design tasks. We find that off-the-shelf reasoning LLMs fail in SMILES-based molecular optimization due to their poor handling of SMILES representations and large in-context inputs, but agentic workflows that leverage cheminformatics tools and statistical model-based filtering overcome these limitations. In contrast, in the design of four-residue protein motifs, pure reasoning LLMs excel by generating domain knowledge-driven hypotheses, while agentic workflows underperform, relying too heavily on tools. These results highlight the complementarity of reasoning models and agentic architectures, offering guidance on when each is preferable. Finally, we show that non-reasoning LLMs trained via supervised fine-tuning (SFT) can efficiently mimic statistical strategies in our setting, sometimes outperforming reasoning models at a fraction of the computational cost. Together, our findings clarify the respective roles and failure modes of reasoning, agentic, and statistical approaches in BO, and propose a path toward hybrid methods that combine the strengths of LLM-hypothesis generation and statistical rigor.

Table 1: Comparison of supervised fine-tuned (SFT), off-the-shelf (LLM), and agentic (Agent) approaches for BO across our two domains.

	Domain			Performance			Relative Cost		
Example	**Representation**	**Search Space**	**Budget**	**SFT**	**LLM**	**Agent**	**SFT**	**LLM**	**Agent**
small AA motifs	Simple	Less constrained	Small	Mid	**High**	Poor	**Low**	Mid	High
Medium-sized SMILES	Complex	Constrained	Large	-	Poor	**High**	-	High	Mid

1 Introduction

Chemical space diversity presents a challenge in scientific discovery, specifically in fields like drug design, protein engineering, and others where the number of potential candidates is prohibitively

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

large to test thoroughly. Bayesian optimization (BO) is a paradigm that addresses the challenge of efficiently navigating large search spaces in an intelligent manner [Snoek et al., 2012]. At its core, BO is a probabilistic machine learning approach where a model iteratively suggests experiments to perform for costly evaluations (e.g., human expert, physical experiment, computational model, etc.). The iterative, data-driven strategy is meant to optimally explore the search space, while exploiting the best possible candidates with minimal number of experiments, thereby accelerating the scientific discovery process [Tabor et al., 2018]. The framework has been successfully used in drug discovery [Korovina et al., 2020, Pyzer-Knapp, 2018], chemical reaction optimization [Shields et al., 2021], catalyst development and more [Tabor et al., 2018, Hsieh et al., 2018].

At the core of the Bayesian optimization framework is the acquisition function (AF). It is the formulation that decides which candidate (or set of) to select at each iteration. Statistical AFs are often designed to manage the fundamental trade-off between exploitation and exploration [Kaelbling et al., 1996, Shahriari et al., 2015]. Exploration will increase the knowledge of the model by focusing on candidates where the model is uncertain, aiming to reduce global error of the model, potentially uncovering novel regions of the search space with better performance. Exploitation, on the other hand, prioritizes already known regions where the model currently predicts the best outcome, effectively optimizing known regions of high performance. Common methods such as Upper Confidence Bound (UCB) [Lai and Robbins, 1985] and Thompson Sampling (TS) [Thompson, 1933, Thompson et al., 2022] offer frameworks for balancing these competing objectives.

The performance of these methods is contingent on the quality of the model's uncertainty estimates and only captures the knowledge from the model predictions, but does not include chemical knowledge available in the large body of literature work. The inherent opacity of the surrogate models typically employed in Bayesian optimization hinders its broader utility, as it precludes researchers from elucidating the fundamental structure-property relationships governing the system. The expensive calculations in common surrogate models such as the Gaussian Process limit the ability of using BO for large search spaces and long optimization campaigns. Collectively, these constraints on interpretability, computational scalability, and knowledge integration limit the practical utility of Bayesian optimization.

Large language models (LLMs) on top of being disruptive in virtually every other field, also bring potential benefits to scientific discovery. The vast domain knowledge and reasoning capabilities of LLMs promise great potential in enhancing BO. Studies have used LLMs for the entire BO pipeline [Yang et al., 2023], replacing representation, uncertainty quantification and acquisition [Wang et al., 2025], or dynamically sampling new proposal distributions as the campaign evolves [Agarwal et al., 2025]. Yet there is a lack of clear understanding as to when and how LLMs are better than classical BO methods.

In this work, we systematically compare LLM-based approaches (off-the-shelf, fine-tuned or agentic) against BO methods. At the core of the investigation, is to understand under what conditions do LLMs perform better than BO, and when not. We do this by using two benchmarks. SMILES-based molecular optimization and a four-residue protein optimization. The SMILES task is characterized by a constrained search-space, where valid SMILES strings form only a small fraction of all possible character sequences, and a large evaluation budget. The protein task is characterized by simple representations (four-residue amino acid sequence) and a less-constrained search space – all amino acid combinations are valid and a small evaluation budget. Our main findings are that off-the-shelf models fail in SMILES-based molecular optimization tasks, due to the inability to accurately understand molecular structure from SMILES strings and failure to process large in-context inputs. Agentic frameworks, that have been equipped with domain specific tools, help to overcome these both failure modes by offloading filtering to tools. In the four-residue protein design task, reasoning models excel due to effective domain related hypothesis generation. On the other hand, agentic workflows under-perform here in comparison to pure reasoning models.

2 Related Works

2.1 Bayesian optimization and active learning in molecular sciences

Bayesian optimization and active learning have been extensively employed in various fields to accelerate scientific discovery. In drug discovery, these methods are used to navigate the vast chemical space to find novel drug candidates, optimizing for properties such as docking scores,

potency, and more [McDonald et al., 2025]. For instance, Dang et al. [2025] showed that BO can guide the synthesis of ligands with high affinity to specific enzymes. In materials science, these techniques have accelerated the discovery of novel liquid electrolytes, identified novel electro-catalyst candidates for CO_2 reduction and oxygen evolution reduction, and discovered new alloys with enhanced mechanical strength [Dave et al., 2022, Jenewein et al., 2024, Tran and Ulissi, 2018, Ghorbani et al., 2024]. Furthermore, BO is frequently applied to optimize the conditions of chemical reactions, efficiently determining the ideal temperature, pressure, and reactant concentrations to maximize yield and minimize byproducts [Tachibana et al., 2023, Burger et al., 2020]. In the field of protein engineering, Bayesian optimization and active learning guide the design of antibodies with high target specificity, proteins with high thermostability, and peptide sequences with favorable functionality and stability [Khan et al., 2023, Stanton et al., 2022, Manshour et al., 2024].

2.2 LLM-guided Bayesian optimization and active learning

With recent advances in LLMs, there is a shift toward the integration of LLMs into the BO/AL loop to exploit their vast prior knowledge and reasoning capabilities. One major direction involves framing the LLM itself as an optimizer [Yang et al., 2023, Xia et al., 2025]. In molecular discovery [Reinhart and Statt, 2024], it has been demonstrated that off-the-shelf LLMs (e.g., Claude 3.5) can serve as an an evolutionary optimizer for macromolecular operations, outperforming traditional active learning pipelines and genetic algorithms on a polymer sequence discovery task, suggesting that LLMs can implicitly balance exploration and exploitation . Similarly, Liu et al. [2024] integrate an LLM into the BO process by framing optimization as a language problem by prompting the model to propose and evaluate solutions given the history of observation. Leveraging the LLM's zero/few- shot learning to guide search, their approach (LLAMBO) improved hyperparameter tuning efficiency in data-sparse settings without any model fine-tuning. Lu et al. [2025] achieve similar results on simple transition metal complexes discovery using in-context learning. However, some studies provide a more cautious perspective. Kristiadi et al. [2024] reports that general-purpose LLMs offer limited benefit, unless they have been pretrained or finetuned with domain-specific data (e.g., molecular data). Additionally, this finding is further corroborated by Wang et al. [2025] who found that no off-the-shelf LLM, no matter incorporated at what stage of BO, can outperform a simple statistical baseline. These results highlight that careful adaptation is needed to leverage LLMs in BO/AL settings. Building on top of these previous works, our work systematically investigate advantages and limitations of using LLMs for BO in scientific discovery, and proposes strategies to overcome the limitations.

2.3 Frameworks: Off-the-shelf vs. fine-tuned vs. agentic LLMs

Scientific research with Large Language Models (LLMs) spans a spectrum from general-purpose application to highly specialized automation. At one end, off-the-shelf models like GPT, Gemini, Claude, Llama, and Qwen serve as powerful, multipurpose tools for tasks such as summarizing literature, generating hypotheses, and writing code, leveraging their vast pretrained knowledge base. Researchers develop finetuned models by further training a base LLM on a specific domain-centric dataset, for example, routes of chemical synthesis or therapeutics to increase performance in specific tasks [Sun et al., 2025, Chaves et al., 2024, Zhang et al., 2025]. The finetuning process hones the model's capabilities, enabling it to generate highly accurate and contextually relevant outputs for niche tasks that a general model would struggle with. Sumers et al. [2023] proposed agentic LLM frameworks that integrate various LLMs into large systems capable of handling more complicated tasks by dividing into smaller tasks, using larger context windows with more advanced memory, and with the ability to perform actions with integrated tools. Agentic frameworks have been successfully utilized in discovering protein design principles [Ghafarollahi and Buehler, 2025b], novel treatments for macular degeneration [Ghareeb et al., 2025], designing novel alloys [Ghafarollahi and Buehler, 2025a], and more [Xia et al., 2025].

3 Overview

We study the success and failure modes of LLMs across two flavors of Bayesian optimization (BO) tasks: SMILES-based molecular optimization and four-residue protein optimization. The molecule task (SMILES-based) is characterized by a constrained search space where valid SMILES strings form only a small fraction of all possible character sequences—and a large evaluation budget. The

protein task (four-residue) is characterized by a less-constrained search space where all amino acid combinations are valid, and a small evaluation budget.

Across both domains, we evaluate three classes of LLM-based methods: (1) off-the-shelf reasoning models, (2) our developed agentic workflow, and (3) non-reasoning LLMs fine-tuned to mimic statistical acquisition strategies. We compare these against standard statistical baselines and analyze where LLM-based approaches succeed or fail.

SMILES-based molecular optimization. For the **molecule** task, we use the benchmark introduced by Gorantla et al. [2024]. The benchmark contains 4 datasets of thousands of medium-sized molecules (95% in the [15, 22] carbon range), with measured binding affinities to a protein target. We run our tests against two of these targets; D2R, and TYK2. The goal is to identify molecules in the top 2% for each target. Following prior work, we initialize with `starting size = 60`, `batch size = 60`, and extend the evaluation budget to 600 (from 360 in the original benchmark) to strengthen the signal. We also evaluate the predictive model's RMSE on the entire database as a metric of how well each acquisition function informs the predictive model.

As statistical baselines, we implement Gaussian process regression with a Tanimoto kernel (4096 bits, radius 2), using four acquisition strategies: Greedy, Upper Confidence Bound (UCB), Thompson Sampling (TS), and Random sampling.

In BO, the effect of starting point is huge. If we start from bad local minima, greedy algorithm would fail miserably. Thus, robustness of the method to the initialization is required. To evaluate robustness, we consider two initialization regimes for the molecule task: (1) starting batches sampled uniformly at random and (2) batches sampled from a single cluster in a UMAP embedding of chemical space.

four-residue protein optimization. For the **protein** task, we adopt the framework of Yang et al. [2025], which provides fitness values for nearly all sequences in the full 20^4 combinatorial search space of two highly epistatic motifs (on GB1 and TrpB). The objective is to discover sequences of maximal fitness.

We evaluate performance using two metrics: (1) maximum fitness score, as in the original benchmark, and (2) recall of top-performing sequences (top 0.5 %), for consistency with the molecule task. Baselines include Greedy, UCB, TS, Random, and a directed evolution strategy (greedy local search around the wildtype). The predictive model is a deep neural network (DNN) ensemble trained on one-hot encoded sequences, as shown to perform best in the original paper. We test both a small campaign (`starting size = 10`, `batch size = 10`, `budget = 60`) and the larger campaign studied by Yang et al. [2025] (`starting size = 96`, `batch size = 96`, `budget = 480`).

4 Off-the-Shelf Reasoning LLMs Struggle with Molecular Optimization

4.1 Setup: Reasoning LLM-based acquisition design and prompting strategy

We evaluate three reasoning models on this benchmark - **Qwen3-32B** [QwenTeam, 2025], **GPT-5-medium reasoning** [OpenAI, 2025], and **Llama-4-maverick-17B** [MetaAI, 2025]. At each iteration, the model receives the campaign description, accumulated observations (SMILES with ground truth fitness values of selected samples), and a candidate pool (SMILES with predictive model fitness values and confidence scores).

The full list of candidates was processed in chunks of size dependent on the context length, selecting a batch from each chunk, and then a final batch from the union of selections from all chunks. Due to context length limits, smaller models (Qwen3, GPT-5) cannot simultaneously attend to all accumulated observations and candidate chunks. For these models, we first prompt to summarize the accumulated observations, and the summary was then used to inform the selections from the chunks. With this method, acquisition takes around 2 minutes and 60,000 tokens per 1,000 samples.

4.2 Poor handling of large in-context inputs and SMILES limits the reasoning model performance.

LLM-based acquisition underperform compared to statistical acquisition functions in terms of recall, effectively mimicking the greedy algorithm in both the random start (Figure 1a), and the bad start

(Figure 1b), showing a limited ability to identify and escape the local minima. Contrary to our expectation that LLMs would leverage scientific reasoning to escape bad starts, their selections rely heavily on GP predictions, providing little additional information gain for the GP model compared with statistical baselines (Figure 1d).

The inferior performance of LLM-based acquisition highlights two main issues. First, LLMs struggle to correctly parse and reason over SMILES strings. Second, they are unable to accurately process the large in-context dataset tables. Qwen3 and Llama-4 often deviate from the required output format, selecting duplicates or using wrong budget. GPT-5 is more precise with the format, but appears to fail at correctly mapping chemical patterns to SMILES (further discussion in A.1). Moreover, finding an optimal way of chunking the data for in-context input requires an extra effort.

5 Agentic Workflows Improve Data Comprehension and Scalability

Given these limitations, we explore whether agentic workflows, where an LLM orchestrates tool use rather than directly ranking molecules, can mitigate data processing challenges. We designed an LLM workflow to select from the candidate pool using a limited set of simple tools, including: (1) sorting by GP predictions, (2) filtering by SMARTS substrings, (3) filtering by Tanimoto similarity in comparison to compounds already observed or to the current batch, and (4) filtering using UCB with hyperparameters set by the agent.

5.1 Setup: Workflow architecture with strategist, implementer, and summary nodes

The workflow begins with a "strategist" node prompted to analyze the campaign stage, objectives, and accumulated observations (a dataframe of SMILES string and score of each molecule selected in previous cycles). Based on this analysis, it proposes a set of selection strategies. Each strategy is then passed to an "implementer" node, responsible for executing the strategy via tool calls. The results, along with the information of strategies, task reasoning, and tool calls, is finally processed by a "summary" node which evaluates the effectiveness of each strategy and the cycle performance overall. This summary is subsequently fed back to the "reasoning" node, informing the design of strategies in the next iteration. Claude-3.5-sonnet [Anthropic, 2024] was used for both implementer and summary nodes.

5.2 Agentic workflows surpass both statistical and LLM baselines

Even with a limited set of tools, the agentic workflow proves effective, greatly surpassing all off-the-shelf models in the easy task (Figure 1a, **AGENT**), and the statistical models on the bad start (Figure 1b). The main reason for this improvement is that the workflow's ability to leverage rule-based SMARTS filtering, rather than having to rely on their own understanding of SMILES.

5.3 Limiting in-context information reinforces domain knowledge-driven reasoning

From the LLM reasoning history, we observed that the agent consistently copied patterns from SMILES present in the accumulated observations table for SMARTS filtering. Interestingly, when we remove all sample-level information (i.e., SMILES strings and their associated scores) from the prompt provided to the "strategist" node (Prompt A-A.4.1, SMILES table at end of cycle summary Prompt A-A.4.1), while maintaining the natural language cycle summaries (generated by the "summary" node), the model engages in expert-level chemical reasoning about structural motifs associated with binding. Instead of copy-pasting substrings from the table of accumulated observations, it instead creates SMARTS filters based on chemical hypotheses from prior knowledge. This method achieves performance comparable to statistical methods on the simpler task (Figure 1a, **SIMPLEAGENT**) while providing larger information gain to the GP model (Figure 1c), and quickly breaks out from the bad starts (Figure 1b). The improvement from the original AGENT suggests that constraining models to tool-mediated reasoning allows domain knowledge to be used more effectively than forcing direct SMILES comprehension.

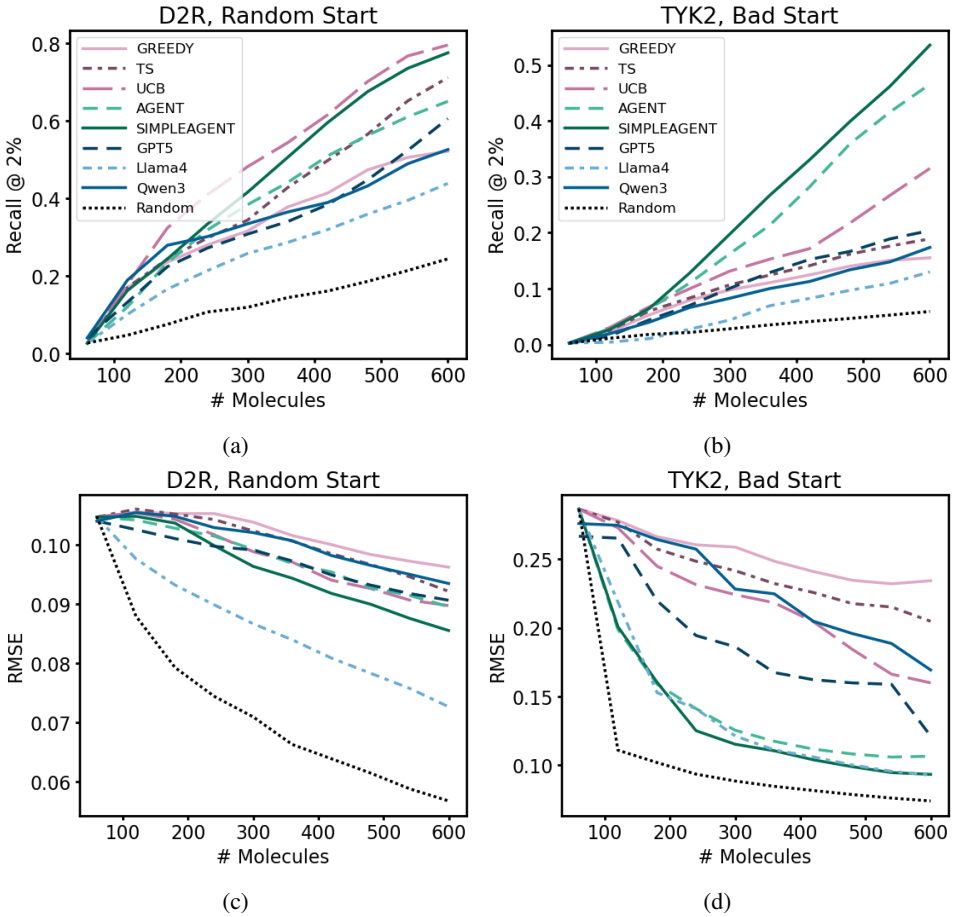


Figure 1: Performance of statistical acquisition functions (**GREEDY**, **UCB** = upper confidence bound, **TS** = Thompson sampling), off-the-shelf models (**GPT-5**, **Qwen3**, **Llama4 (non-reasoning)**), and agentic flows (**AGENT**, **SIMPLEAGENT**) on the molecular BO task. Metrics: (1) Recall at 2 % (**Top row**), the fraction of the top 2 % of candidates in the entire search space, (2) predictive model's RMSE on the accumulated observations (**Bottom row**) measuring how well-informed the predictive model is of the search space at a given stage of the campaign. **Left column:** Binding affinity optimization for D2R target with random initialization. **Right column:** Binding affinity optimization for TYK2 target, with initialization from a bad local minima testing the method's ability to explore. N=10 samples for all methods except Qwen3-D2R random start (N=5), Qwen3-TYK2 bad start (N=5), GPT-5-D2R random start (N=5), and GPT-5-TYK2 bad start (N=4). Dashed line (**Random**) marks the performance of random acquisition. Agentic models outperform statistical models when the campaign is initialized in a bad local minima and are competitive on random initializations, while informing the predictive model more.

6 Protein optimization task shows the true power of reasoning model

From our previous section, we identified the failure modes in SMILES-based molecular optimization task. LLMs are not good at comprehending SMILES and in-context processing of large dataset is not ideal. As such, we next investigated whether these models are more effective in the task with simpler representations and less-constrained search spaces, optimizing the fitness of a four-position protein motif.

The set of current observations for statistical models and the agentic workflows were initialized with random samples from the search space and a target protein description that included the wildtype (WT) sequence. The "off-the-shelf" reasoning models were given the same background and WT. The

agentic workflow could filter by predictive model predictions, blosum62 scores, Hamming distances, regex patterns, and UCB.

6.1 Qwen3 agentic workflows collapse to directed-evolution behavior and underperform

Across both protein targets GB1 and TrpB, Qwen3 underperformed across all metrics (Figure 2). Closer inspection on the reasoning log revealed that its proposals were almost exclusively point mutations of the WT or current best-performing sequence, effectively reproducing directed evolution (DE) (Figure 2, DE baseline).

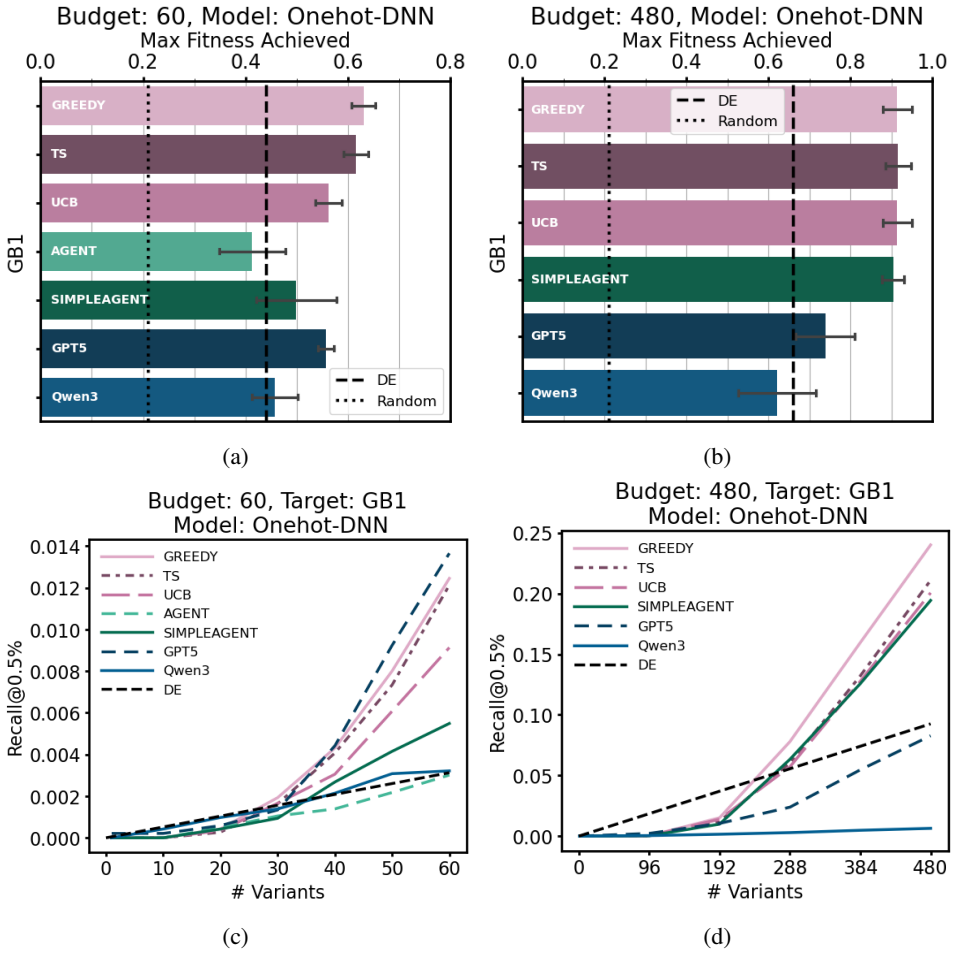


Figure 2: Performance of statistical acquisition functions (**GREEDY**, **UCB** = upper confidence bound, **TS** = Thompson sampling), off-the-shelf models (**GPT-5**, **Qwen3**), and Qwen3-based agentic models (**AGENT**, **SIMPLEAGENT**) on the active learning task of optimizing a four-residue motif. Metrics: (1) highest achieved fitness (**Top row**), (2) recall at 0.5 % (**Bottom row**), implying the fraction of the top 0.5 % motifs in the entire search space. The campaign is performed on two scales: with a budget of 60, batch size 10 motifs (**Left column**), and with a budget of 480, batch size 96 motifs (**Right column**). Error bars mark standard deviation. Statistical methods use 50 samples, and LLM methods use 10 samples. Dashed line (**DE**) marks the performance of directed evolution starting from the WT. Dotted line (**Top row, Random**) marks expected performance from random acquisition. GPT-5 performs the best in the small campaign, but performance deteriorates with increased batch sizes and budgets. Agents fail due to the limited usefulness of tools in this domain.

The Qwen3-based agent also performed poorly: it heavily relied on regex-matching of the accumulated observations and again converged to DE-like behavior (Figure 2a). To mitigate this, we limited the information provided to the initial prompt, withholding any information about the sequences in the accumulated observations so far. This SIMPLEAGENT variant showed marked improvement (Figure

2a), as it relied more on the GP model predictions rather than trying to search near the previous observations. Nevertheless, its recall remained substantially lower than statistical baselines (Figure 2d).

Although performing significantly worse than the statistical models on the recall metric (Figure 2d), it still scales indefinitely in execution time and cost with budget. Extensive tool tuning, or unlimited accessibility to python would likely improve performance significantly.

6.2 Off-the-shelf GPT-5 outperforms statistical methods with effective hypothesis discovery in the smaller campaign

We next evaluated GPT-5, which produced highly potent results in the protein task. Starting from the WT, GPT-5 identified 11 sequences in the top 0.5% of the full search space within just 60 evaluations (Figure 2a). Notably, GPT-5 appeared to have biological knowledge about the system: given only the WT sequence "VDGV" along with the keyword "epistatic," it identified GB1 as the protein target.

We therefore ran both Qwen3 and GPT-5 without any context about the wildtype or the protein, and still observed very strong performance of GPT-5. Both Qwen3 and GPT-5 demonstrated explicit hypothesis-driven search. While Qwen3 still appeared ineffective at executing on the ideas, GPT-5 had already by the second cycle, articulated and tested the hypothesis that "bulky residues at positions 1 and 3, small residues elsewhere" yield high-fitness sequences, a pattern that is 5.3x more common in the top 0.5% of sequences than on average. Moreover, GPT-5 generated multiple hypotheses per batch, enabling rapid exploration of the sequence landscape, in sharp contrast to Qwen3's incremental point mutations. However, GPT-5 suffered from scaling limitations. In larger campaigns with high batch sizes, its performance plateaued (Figures 2b, 2d).

6.3 Generative reasoning scales efficiently independent of search space size

An important note is that the off-the-shelf models are fully generative, producing new sequences directly rather than evaluating an entire candidate pool. This makes their runtime independent of the search space size. Thompson sampling, which requires repeated retraining of the predictive model and covariance matrix computations scales by $O(N^3)$ when using an exact GP in a search space of N candidates, and $O(Nm)$ using low-rank (m rank) to approximate the covariance matrix. This efficiency advantage suggests a role for reasoning models in settings where computational overhead dominates.

7 Fine-tuning Non-Reasoning LLMs Trains the Models to Perform Bayesian Acquisition Behavior

As a complement to the other findings, we aimed to see if non-reasoning Qwen2.5-7B-Instruct could be fine-tuned to perform simpler generative tasks. Off-the-shelf Qwen2.5-7B-Instruct is unable to effectively perform the task. Wang et al. [2025] showed that training a non-reasoning LLM on acquisitions by a statistical model in an artificial setting using Direct Preference Optimization (DPO) can improve the Bayesian behavior of the LLM on selection tasks very similar to the molecular task described in this article.

We similarly generated synthetic AL tasks in the motif domain, and trained the non-reasoning LLM using DPO and supervised fine-tuning (SFT) using trajectories sampled from onehot-DNN Ensemble-TS method as training examples. DPO underperformed in this task, but SFT significantly outperformed Qwen3, and achieved performance competitive with the statistical methods (details in the Appendix A.2.5).

8 Conclusion

In this paper, various LLM-based methods (off-the-shelf, agentic and fine-tuned) are systematically compared on two Bayesian optimization tasks. Models of various sizes and capabilities are compared to gain insight into the scope and potential of using LLMs for BO.

SMILES-based molecular optimization gives enhanced perspective on a common problem for LLMs: off-the-shelf models struggle with in-context processing of large datasets and accurate parsing of

Table 2: Comparison of LLM approaches for Active Learning across different domains

(a) SMILES-Based Molecular Optimization

Method	TYK2, Bad Start		D2R, Random Start	
	Recall@2% ↑	RMSE ↓ $\times 10^{-3}$	Recall@2% ↑	RMSE ↓ $\times 10^{-3}$
Best Statistical Model	0.32 ± 0.25	160 ± 47	0.80 ± 0.05	90 ± 5
Best Agent	$\mathbf{0.54 \pm 0.12}$	$\mathbf{93 \pm 7}$	0.78 ± 0.05	$\mathbf{86 \pm 6}$
Best Off-the-Shelf	0.20 ± 0.18	$\mathbf{121 \pm 13}$	0.61 ± 0.09	91 ± 5

(b) four-residue Protein Optimization

Method	GB1, Budget 60		TrpB, Budget 60	
	Recall@0.5% ↑ $\times 10^{3}$	Max Fitness ↑	Recall@0.5% ↑ $\times 10^{3}$	Max Fitness ↑
Best Statistical Model	12 ± 7	0.63 ± 0.17	9 ± 10	0.51 ± 0.32
Best Agent	5 ± 6	0.50 ± 0.25	7 ± 7	0.57 ± 0.20
Best Off-the-Shelf	$\mathbf{20 \pm 11}$	0.60 ± 0.13	10 ± 15	0.45 ± 0.35

SMILES strings. The introduction of an agentic workflow with the ability to use a set of simple tools greatly increases the performance of LLMs on this task. Furthermore, by removing SMILES related information from the agent the model performance increases even more. This happens as the model creates chemical hypothesis more freely and relies less on direct SMILES string comprehension. Table 2 (a) shows the performance increase of the best agent framework in this task.

The protein optimization task shows the power of reasoning models on simpler representations. While the smaller models (Qwen3-32B) and the agentic frameworks saw performance much lower than that of the BO baselines, the GPT-5 reasoning model generated a wide variety of valid hypothesis and rapidly explored the sequence landscape. On the other hand, GPT-5 suffered from scaling to larger campaigns with high batch sizes, where its performance plateaued. Finally, fine-tuned non-reasoning models were able to achieve enhanced performance compared to their non-fine-tuned reasoning counterparts while operating on a fraction of cost.

This study shows the potential for using LLMs in BO in various fields under the correct circumstances. In constrained search spaces such as SMILES optimization, agentic workflows equipped with external tools yield stronger performance, whereas in less-constrained search spaces with more LLM-comprehensible representations such as protein optimization, off-the-shelf reasoning models employing generative strategies prove more effective.

References

Dhruv Agarwal, Manoj Ghuhan Arivazhagan, Rajarshi Das, Sandesh Swamy, Sopan Khosla, and Rashmi Gangadharaiah. Searching for optimal solutions with LLMs via bayesian optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=aVfDrl7xDV`.

Anthropic. Claude 3.5 sonnet model card addendum. `https://www.anthropic.com/news/claude-3-5-sonnet`, 2024. Accessed August 2025.

Benjamin Burger, Phillip M Maffettone, Vladimir V Gusev, Catherine M Aitchison, Yang Bai, Xiaoyan Wang, Xiaobo Li, Ben M Alston, Buyi Li, Rob Clowes, et al. A mobile robotic chemist. *Nature*, 583(7815):237–241, 2020.

Juan Manuel Zambrano Chaves, Eric Wang, Tao Tu, Eeshit Dhaval Vaishnav, Byron Lee, S Sara Mahdavi, Christopher Semturs, David Fleet, Vivek Natarajan, and Shekoofeh Azizi. Tx-llm: A large language model for therapeutics. *arXiv preprint arXiv:2406.06316*, 2024.

Tai Dang, Long-Hung Pham, Sang T Truong, Ari Glenn, Wendy Nguyen, Edward A Pham, Jeffrey S Glenn, Sanmi Koyejo, and Thang Luong. Preferential multi-objective bayesian optimization for drug discovery. *arXiv preprint arXiv:2503.16841*, 2025.

Adarsh Dave, Jared Mitchell, Sven Burke, Hongyi Lin, Jay Whitacre, and Venkatasubramanian Viswanathan. Autonomous optimization of non-aqueous li-ion battery electrolytes via robotic experimentation and machine learning coupling. *Nature communications*, 13(1):5454, 2022.

Alireza Ghafarollahi and Markus J Buehler. Automating alloy design and discovery with physics-aware multimodal multiagent ai. *Proceedings of the National Academy of Sciences*, 122(4): e2414074122, 2025a.

Alireza Ghafarollahi and Markus J Buehler. Sparks: Multi-agent artificial intelligence model discovers protein design principles. *arXiv preprint arXiv:2504.19017*, 2025b.

Ali Essam Ghareeb, Benjamin Chang, Ludovico Mitchener, Angela Yiu, Caralyn J Szostkiewicz, Jon M Laurent, Muhammed T Razzak, Andrew D White, Michaela M Hinks, and Samuel G Rodriques. Robin: A multi-agent system for automating scientific discovery. *arXiv preprint arXiv:2505.13400*, 2025.

Marzie Ghorbani, M Boley, PNH Nakashima, and Nick Birbilis. An active machine learning approach for optimal design of magnesium alloys using bayesian optimisation. *Scientific Reports*, 14(1): 8299, 2024.

Rohan Gorantla, Alžbeta Kubincová, Benjamin Suutari, Benjamin P. Cossins, and Antonia S. J. S. Mey. Benchmarking active learning protocols for ligand-binding affinity prediction. *Journal of Chemical Information and Modeling*, 64(6):1234–1256, 2024. doi: 10.1021/acs.jcim.4c00220. URL `https://pubs.acs.org/doi/10.1021/acs.jcim.4c00220`.

Hsiao-Wu Hsieh, Connor W Coley, Lorenz M Baumgartner, Klavs F Jensen, and Richard I Robinson. Photoredox iridium–nickel dual-catalyzed decarboxylative arylation cross-coupling: from batch to continuous flow via self-optimizing segmented flow reactor. *Organic process research & development*, 22(4):542–550, 2018.

Ken J Jenewein, Luca Torresi, Navid Haghmoradi, Attila Kormányos, Pascal Friederich, and Serhiy Cherevko. Navigating the unknown with ai: multiobjective bayesian optimization of non-noble acidic oer catalysts. *Journal of Materials Chemistry A*, 12(5):3072–3083, 2024.

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

Asif Khan, Alexander I Cowen-Rivers, Antoine Grosnit, Derrick-Goh-Xin Deik, Philippe A Robert, Victor Greiff, Eva Smorodina, Puneet Rawat, Rahmad Akbar, Kamil Dreczkowski, et al. Toward real-world automated antibody design with combinatorial bayesian optimization. *Cell Reports Methods*, 3(1), 2023.

Ksenia Korovina, Sailun Xu, Kirthevasan Kandasamy, Willie Neiswanger, Barnabas Poczos, Jeff Schneider, and Eric Xing. Chembo: Bayesian optimization of small organic molecules with synthesizable recommendations. In *International Conference on Artificial Intelligence and Statistics*, pages 3393–3403. PMLR, 2020.

Agustinus Kristiadi, Felix Strieth-Kalthoff, Marta Skreta, Pascal Poupart, Alán Aspuru-Guzik, and Geoff Pleiss. A sober look at llms for material discovery: Are they actually good for bayesian optimization over molecules? *arXiv preprint arXiv:2402.05015*, 2024.

T. L. Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985. doi: 10.1016/0196-8858(85)90002-8.

Tennison Liu, Nicolás Astorga, Nabeel Seedat, and Mihaela van der Schaar. Large language models to enhance bayesian optimization. *arXiv preprint arXiv:2402.03921*, 2024.

Jieyu Lu, Zhangde Song, Qiyuan Zhao, Yuanqi Du, Yirui Cao, Haojun Jia, and Chenru Duan. Generative design of functional metal complexes utilizing the internal knowledge and reasoning capability of large language models. *Journal of the American Chemical Society*, 2025. doi: 10.1021/jacs.5c02097.

Negin Manshour, Fei He, Duolin Wang, and Dong Xu. Integrating protein structure prediction and bayesian optimization for peptide design. *Research Square*, pages rs–3, 2024.

Matthew A McDonald, Brent A Koscher, Richard B Canty, Jason Zhang, Angelina Ning, and Klavs F Jensen. Bayesian optimization over multiple experimental fidelities accelerates automated discovery of drug molecules. *ACS central science*, 11(2):346–356, 2025.

MetaAI. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. `https://ai.meta.com/blog/llama-4-multimodal-intelligence/`, 4 2025. Accessed August 2025.

OpenAI. Gpt-5 system card. `https://cdn.openai.com/gpt-5-system-card.pdf`, 8 2025. Accessed August 2025.

Edward O Pyzer-Knapp. Bayesian optimization for accelerated drug discovery. *IBM Journal of Research and Development*, 62(6):2–1, 2018.

QwenTeam. Qwen3 technical report. `https://arxiv.org/abs/2505.09388`, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Chelsea Finn, and Stefano Ermon. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, cs.LG, jul 2024. URL `https://arxiv.org/abs/2305.18290`. 37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506, New York, NY, USA, 2020. Association for Computing Machinery. doi: 10.1145/3394486.3406703. URL `https://dl.acm.org/doi/10.1145/3394486.3406703`. Tutorial.

Wesley F Reinhart and Antonia Statt. Large language models design sequence-defined macromolecules via evolutionary optimization. *npj Computational Materials*, 10(1):262, 2024.

Nicholas Runcie. Gpt-5 achieves state-of-the-art chemical intelligence. `https://www.blopig.com/blog/2025/08/gpt-5-achieves-state-of-the-art-chemical-intelligence/`, August 2025. Oxford Protein Informatics Group Blog.

Nicholas T. Runcie, Charlotte M. Deane, and Fergus Imrie. Assessing the chemical intelligence of large language models. *arXiv preprint arXiv:2505.07735*, v2, July 2025. URL `https://arxiv.org/abs/2505.07735`.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2015.

Benjamin J Shields, Jason Stevens, Jun Li, Marvin Parasram, Farhan Damani, Jesus I Martinez Alvarado, Jacob M Janey, Ryan P Adams, and Abigail G Doyle. Bayesian reaction optimization as a tool for chemical synthesis. *Nature*, 590(7844):89–96, 2021.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Samuel Stanton, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. Accelerating bayesian optimization for biological sequence design with denoising autoencoders. In *International conference on machine learning*, pages 20459–20478. PMLR, 2022.

Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2023.

Kunyang Sun, Dorian Bagni, Joseph M Cavanagh, Yingze Wang, Jacob M Sawyer, Andrew Gritsevskiy, Oufan Zhang, and Teresa Head-Gordon. Synllama: Generating synthesizable molecules and their analogs with large language models. *arXiv preprint arXiv:2503.12602*, 2025.

Daniel P Tabor, Loïc M Roch, Semion K Saikin, Christoph Kreisbeck, Dennis Sheberla, Joseph H Montoya, Shyam Dwaraknath, Muratahan Aykol, Carlos Ortiz, Hermann Tribukait, et al. Accelerating the discovery of materials for clean energy in the era of smart automation. *Nature reviews materials*, 3(5):5–20, 2018.

Ryo Tachibana, Kailin Zhang, Zhi Zou, Simon Burgener, and Thomas R Ward. A customized bayesian algorithm to optimize enzyme-catalyzed reactions. *ACS Sustainable Chemistry & Engineering*, 11 (33):12336–12344, 2023.

James Thompson, W Patrick Walters, Jianwen A Feng, Nicolas A Pabon, Hongcheng Xu, Michael Maser, Brian B Goldman, Demetri Moustakas, Molly Schmidt, and Forrest York. Optimizing active learning for free energy calculations. *Artificial Intelligence in the Life Sciences*, 2:100050, 2022. doi: 10.1016/j.ailsci.2022.100050. URL https://www.sciencedirect.com/science/article/pii/S2667318522000204.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933. doi: 10.1093/biomet/25.3-4.285.

Kevin Tran and Zachary W Ulissi. Active learning across intermetallics to guide discovery of electrocatalysts for co2 reduction and h2 evolution. *Nature Catalysis*, 1(9):696–703, 2018.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Yuanqing Wang, Yuzhi Xu, Stefano Martiniani, Theofanis Karaletsos, Andrew Gordon Wilson, and Kyunghyun Cho. Molecular active learning: How can llms help? *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=kYg04pmX7i. Withdrawn submission.

Yu Xia, Subhojyoti Mukherjee, Zhouhang Xie, Junda Wu, Xintong Li, Ryan Aponte, Hanjia Lyu, Joe Barrow, Hongjie Chen, Franck Dernoncourt, Branislav Kveton, Tong Yu, Ruiyi Zhang, Jiuxiang Gu, Nesreen K. Ahmed, Yu Wang, Xiang Chen, Hanieh Deilamsalehy, Sungchul Kim, Zhengmian Hu, Yue Zhao, Nedim Lipka, Seunghyun Yoon, Ting-Hao 'Kenneth' Huang, Zichao Wang, Puneet Mathur, Soumyabrata Pal, Koyel Mukherjee, Zhehao Zhang, Namyong Park, Thien Huu Nguyen, Jiebo Luo, Ryan A. Rossi, and Julian McAuley. From selection to generation: A survey of llm-based active learning, May 2025. URL https://arxiv.org/pdf/2502.11767.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*, 2023.

Kevin Y Yang, Kadina Swanson, Wengong Jin, Connor W Coley, Pieter Abbeel, and Regina Barzilay. Active learning-assisted directed evolution. *Nature Communications*, 16:1089, 2025.

Yutong Zhang, Yujie Wang, Yiming Zhang, Yutong Wang, Yichi Zhang, Yuxuan Wang, Yifan Wang, Yuxuan Zhang, Yuxin Zhang, Yuxin Wang, Yuxiang Wang, Yuxiang Zhang, Yuxiao Wang, and Yuxiao Zhang. Generative molecule design with large language models: A benchmarking study. *arXiv preprint arXiv:2501.19153*, 2025.

Ziqiao Zhang, Bangyi Zhao, Ailin Xie, Yatao Bian, and Shuigeng Zhou. Activity cliff prediction: Dataset and benchmark. *arXiv preprint arXiv:2302.07541*, 2023. URL https://arxiv.org/abs/2302.07541.

A Technical Appendices and Supplementary Material

Method	TYK2 Bad Start		TYK2 Random Start		D2R Bad Start		D2R Random Start	
	Recall@2% ↑	RMSE ↓ $\times 10^{-3}$	Recall@2% ↑	RMSE ↓ $\times 10^{-3}$	Recall@2% ↑	RMSE ↓ $\times 10^{-3}$	Recall@2% ↑	RMSE ↓ $\times 10^{-3}$
GREEDY	0.16 ± 0.14	234 ± 20	0.62 ± 0.04	100 ± 4	0.47 ± 0.17	123 ± 9	0.52 ± 0.13	96 ± 6
UCB	0.32 ± 0.25	160 ± 47	$\mathbf{0.69 \pm 0.04}$	94 ± 4	0.68 ± 0.08	110 ± 7	0.80 ± 0.05	90 ± 5
TS	0.19 ± 0.15	205 ± 41	0.65 ± 0.04	99 ± 5	0.56 ± 0.15	118 ± 9	0.71 ± 0.07	92 ± 6
AGENT	$\mathbf{0.47 \pm 0.14}$	$\mathbf{107 \pm 33}$	0.62 ± 0.11	$\mathbf{90 \pm 4}$	0.70 ± 0.09	$\mathbf{99 \pm 7}$	0.65 ± 0.10	90 ± 6
SIMPLEAGENT	$\mathbf{0.54 \pm 0.12}$	$\mathbf{93 \pm 7}$	0.64 ± 0.06	$\mathbf{86 \pm 2}$	0.69 ± 0.08	$\mathbf{98 \pm 10}$	0.78 ± 0.05	$\mathbf{86 \pm 6}$
GPT-5	0.20 ± 0.18	$\mathbf{121 \pm 13}$	-	-	-	-	0.61 ± 0.09	91 ± 5
Qwen3	0.17 ± 0.17	169 ± 45	-	-	-	-	0.53 ± 0.10	93 ± 6

Table 3: Performance comparison for different acquisition methods on the molecular domain. Errors are standard deviations (N=4 GPT-5 TYK2 bad start; N=5 GPT other; N=5 Qwen3; N=10 all others). Bold numbers mark significantly better performance in a statistical method relative to all LLM-method or an LLM-method relative to all statistical method (95% confidence in difference between means by bootstrapping on random seeds).

Method	Budget 60, GB1		Budget 480, GB1		Budget 60, TrpB	
	Recall@0.5% ↑ $\times 10^3$	Max Fitness ↑	Recall@0.5% ↑ $\times 10^3$	Max Fitness ↑	Recall@0.5% ↑ $\times 10^3$	Max Fitness ↑
GREEDY	12 ± 7	0.63 ± 0.17	$\mathbf{231 \pm 60}$	0.89 ± 0.13	9 ± 10	0.51 ± 0.32
UCB	9 ± 7	0.56 ± 0.18	196 ± 53	0.90 ± 0.12	7 ± 8	0.50 ± 0.30
TS	12 ± 8	0.62 ± 0.17	216 ± 55	0.90 ± 0.12	9 ± 9	0.52 ± 0.29
AGENT	-	-	-	-	-	-
SIMPLEAGENT	5 ± 6	0.50 ± 0.25	194 ± 52	0.90 ± 0.08	7 ± 7	0.57 ± 0.20
GPT5	14 ± 7	0.56 ± 0.05	83 ± 23	0.74 ± 0.18	$\mathbf{16 \pm 9}$	0.58 ± 0.05
Qwen3	3 ± 3	0.46 ± 0.14	6 ± 3	0.60 ± 0.20	11 ± 8	$\mathbf{0.61 \pm 0.08}$
Qwen3-BLIND	3 ± 4	0.41 ± 0.19	-	-	3 ± 5	0.43 ± 0.32
GPT5-BLIND	$\mathbf{20 \pm 11}$	0.60 ± 0.13	-	-	10 ± 15	0.45 ± 0.35

Table 4: Performance comparison for different acquisition methods on the protein domain. Errors mark standard deviations (N=50 for statistical models, N=10 otherwise). Bold numbers mark significantly better performance in a statistical method relative to all LLM-method or an LLM-method relative to all statistical method (95% confidence by bootstrapping on random seeds for agent-statistical comparisons (N=10), and on unique trajectories on off-the-shelf LLM (N=10)-statistical (N=50) comparisons). Note: Averages and marginal errors shown for statistical models here are across 50 samples, but significance between agents and statistical models is still computed on 10 shared starting points.

A.1 LLMs have trouble comprehending SMILES

In depth analysis of the off-the-shelf LLM responses showed several failure modes. Firstly, the Qwen3 and Llama (and Claude-Sonnet-3.7, not included in data) were greedy with respect to selecting SMILES strings. When prompted with a table sorted by predicted affinity, they exclusively chose from the very top of the table. When the table was shuffled, they sometimes appeared to only process subsets of the table. As seen in Response A.1, Llama-4-maverick has a very high probability of selecting a candidate index following a candidate index it has already selected. This is not a surprising, but a serious issue. The reasoning was often advanced, but despite extensive prompt engineering efforts, LLMs struggled to implement their stated strategies in practice. Models would provide excellent rationales for decisions and develop sophisticated AL strategies during the reasoning phase, yet default to simple heuristics during selection. The reasoning around specific molecule structures was often extremely brief and related only to one or two substructures, often on the edges of the SMILES string (Figure 3, right). The model often confused different substructures in multi-ring systems. To further investigate SMILES comprehension, we asked Qwen3 to summarize different SMILES from our database (Figure 3, left). The summaries often include several correctly named substructures and correct chemical properties, but the relation between the substructures is absent. Qwen3 also often uses SMILES substrings to design hypothesis and filters. This leads to very dense summaries of the chemical space largely containing copy-pasted substrings (Response A.4.1). This shows that the reasoning models are capable of using substructre information from the SMILES, but that it requires significant token use. When prompted with more SMILES, precision decreases drastically. Llama-4-maverick reasons in a similar fashion. Qwen3 and Llama-4, rely heavily on overly simple filters that only capture minimal information about a given molecule's performance.

Other studies have similarly shown that reducing detailed numeric or conceptually complex data increases performance [Agarwal et al., 2025].

Interestingly, GPT5 uses a completely different language. The summaries are more extensive (Response A.4.1) and detailed. It is able to derive relations between substructures, for example *"Substructure filter: pyridyl-diamide with two secondary amides: Ar$\hat{L}$–NH–C(=O)–pyridyl–NH–C(=O)–Ar$\hat{R}$"*. GPT5's failure likely lies in its ability to apply such derived filters in practice. We observed several instances in which it was mapping even simple substructures incorrectly, stating that it found sulfonyl in the indices of molecules without a single sulfur. GPT5 has indeed been shown to perform better than other models on atom mapping, which may translate to better understanding relations between substructures, but not on SMILES to IUPAC [Runcie, 2025, Runcie et al., 2025]. We aim to investigate this further in subsequent work.

GPT actually recognizes that the chemical space is constrained in the clustered starting configurations (Figure 1b) but does nothing about it: `"The chemical space is tightly focused around a single core with well-behaved, monotonic SAR along three modular regions."`

Response 1: Llama-4-maverick response from processing chunk

```
...
Here are 60 selected indices, ensuring a mix across the entire range and
    diversity in chemical structures and predicted properties:

<selected_indices>
[38, 497, 244, 248, 293, 294, 999, 1000, 113, 116, 140, 599, 605, 758, 762,
    838, 839, 847, 851, 884, 885, 937, 940, 949, 950, 951, 952, 953, 954, 10,
     20, 31, 32, 33, 34, 39, 48, 52, 60, 62, 70, 106, 108, 115, 117, 118,
    119, 121, 122, 123, 124, 125, 130, 131, 132, 133, 134, 135, 136, 137,
    138, 139]
</selected_indices>
```

A.2 Implementation Details

A.2.1 Bayesian Optimization Loop

Each Bayesian Optimization campaign is characterized by an `initial size` of set of observations selected randomly or from a cluster, a `batch size` denoting the number of samples to be selected or generated each loop, and a `budget` of total samples drawn throughout the campaign. The number of cycles is defined directly from the `batch size` and the `budget`. Each cycle follows the following scheme:

1. Train a `predictive model` on accumulated observations
2. Use the `predictive model` to predict the fitness of all candidates in the search space
3. Sample observations from the predictions using an `Acquisition function`
4. Assign true labels to all sampled observations and add them to the set of accumulated observations

A.2.2 LLM-based approaches

This project investigates the use of Large Language Models (LLMs) as the predictive model and/or acquisition function within the BO loop. We compare the performance of three distinct LLM strategies in this role:

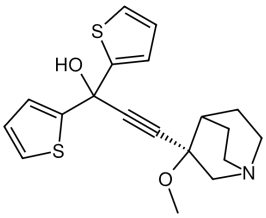
- **Off-the-shelf model:** A base LLM applied directly using zero-shot or few-shot prompting. For the molecular optimization task the LLM replaces only the acquisition function. For the protein optimization task the LLM replaces both the predictive model and the acquisition function and is completely standalone.
- **Fine-tuned model:** An LLM finetuned to the specific optimization task through further training on a domain-specific dataset. The fine-tuned model was trained only for the protein optimization task.
- **Agentic workflow:** A system where an LLM orchestrates a more complex, multi-step reasoning process to sample the candidate space. The agentic workflow replaces the acquisition function.

The project used various publicly available LLM families. Llama-4-maverick [MetaAI, 2025] and Qwen3-32B [QwenTeam, 2025] which were accessed through the Lambda API. GPT-5 **?** was accessed through the OpenAI API. Claude Anthropic [2024] models were accessed through the Anthropic API. Local Qwen models were accessed from Huggingface.

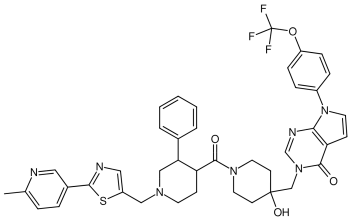
A.2.3 Molecular optimization task

The project utilized publicly available binding affinity datasets assembled by Gorantla et al. [2024]; **TYK2**: 9,997 congeneric molecules with aminopyrimidine core scaffold, derived from RBFE calculations [Thompson et al., 2022], **USP7**: 4,535 diverse scaffolds from ChEMBL, exhibiting multiple assay minima, **D2R**: 2,502 molecule subset of ACNet dataset [Zhang et al., 2023], high activity cliff content, **Mpro**: COVID Moonshot project data, smallest dataset (665 compounds). The target Mpro was discarded because of insufficient data, and USP7 was discarded due to irregularities in the data. The TYK2 search space consists of several larger clusters of molecules, making the search trivial when initialized randomly. The D2R search space consists of a diverse set of molecules. We therefore mainly used TYK2 data to initialize a bad start from one of the clusters, and the D2R data for random starts. Bad start was initialized by clustering (K-means, 10 clusters) a UMAP of the chemical space (10 components), and drawing all initial observations from the same cluster. For completion, we also ran TYK2 with random start and D2R with bad starts for a subset of the methods tested.

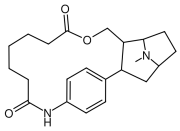
A Gaussian process regressor (GP) with a Tanimoto similarity kernel (radius=4, nbits=4096) trained for 500 epochs (`learning rate=0.001, Adam, gpytorch 1.14, rdkit 2025.3.3`) was used as a predictive model. The GP generates both predictions and their corresponding standard deviations. A random selector drew samples from the candidates from a uniform distribution. UCB directly



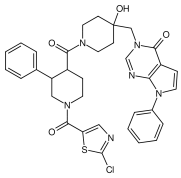
"Carbonyl (CO), thiol groups (two -SCH2-), hydroxyl (terminal -OH), alkyne (C#C). Five-membered heterocyclic ring (piperidine derivative) with chiral center. Alkyne is rigid; rest of molecule flexible. H-bond donors: hydroxyl; acceptors: carbonyl. Moderate size (∼15 heavy atoms). Complex structure with multiple stereocenters and functionalgroups." ✓



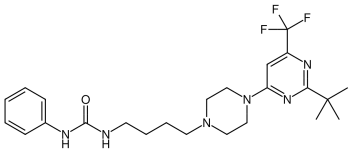
"...thiazole-piperidine with trifluoromethoxy group.." ✓



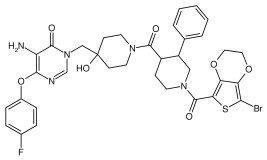
"Pyrrolidine ring (heterocyclic), benzene ring (aromatic), ester (C=O-O), amide (C=O-NH). Flexible backbone with multiple rotatable bonds (ester, amide, alkyl chains). H-bond donors: amide NH, pyrrolidine NH. Acceptors: amide carbonyl, ester carbonyl, ester O. Medium size, complex structure with 4 rings and 5 functional groups." ✓



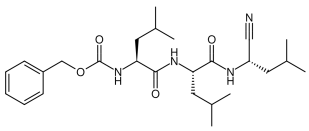
"...chlorothiazole variant with high confidence..." ✓



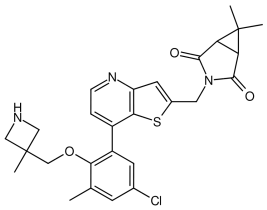
Key functional groups: isopropyl, piperidine (heterocyclic) ×, benzene (aromatic), carbonyl, amide, trifluoromethyl. Ring systems: six-membered piperidine, aromatic benzene. Flexibility: multiplerotatable bonds (amide, piperidine, side chains). H-bonding: amide NH (donor), carbonyl (acceptor), piperidine N (donor/acceptor). Size: ∼20 heavy atoms ×, moderate complexity. Cationic: piperidine N may protonate at physiological pH. Hydrophobic: trifluoromethyl, aromatic rings; hydrophilic: amide, piperidine.



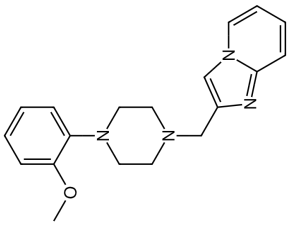
"...brominated benzodioxole, unique substitution pattern..." ×



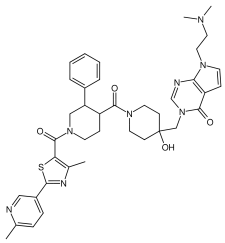
"Tripeptide with nitrile (C#N), isopropyl, and phenyl side chains. Aromatic benzene ring. Multiple rotatable bonds (peptide backbone, side chains). Amide NH donors (3) and carbonyl acceptors (3). Ester O (acceptor). Chiral centers (3). ∼18 heavy atoms ×. Cationic at physiological pH (nitrile, amides). Hydrophobic (isopropyl, phenyl) and hydrophilic (amide, ester) regions.



"...cyclopropylmethyl variant of successful scaffold..." ×



"Aromatic benzene ring with methoxy group (OCH_3) at C1. Piperazine (six-membered, two N) and pyrrolidine (five-membered, one N) heterocycles. Flexibility from ring conformations and single bonds. H-bond donors: piperazine NH, pyrrolidine NH ×; acceptors: methoxy O. Moderate size ∼15 heavy atoms ×), complex structure with multiplerings."



"...with a dimethylaminoethyl group..." ✓

Figure 3: **Left:** Qwen3 responses when asked to summerize given SMILES. **Right:** Qwen3 reasoning about given SMILES during selection or chemical space summary.

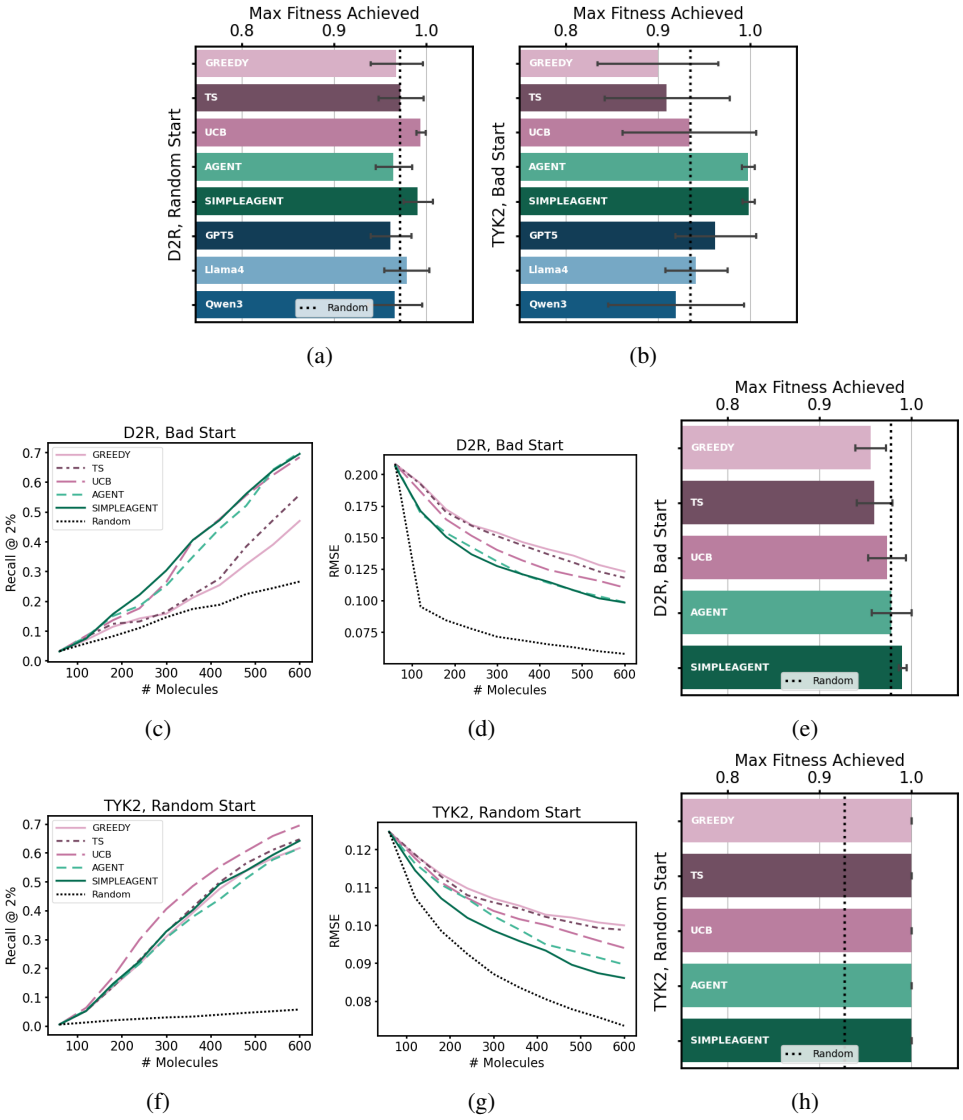


Figure 4: **Top row:** Complement to Figure 1, max fitness achieved in D2R campaign with random start (**left**) and in TYK2 campaign with bad start (**right**). **Middle:** Campaign results from D2R campaign with bad start. **Bottom row:** Campaign results from TYK2 with random start. Error bars mark standard deviation. The SIMPLEAGENT achieves comparable results to the best statistical methods (UCB, TS) (**4c, 4f**) while informing the predictive model more (**4d, 4g**), and often discovering higher affinity molecules within the budget.

sampled the top candidates from the score $s = \mu + \beta^{1/2}\sigma$, using $\beta = 4$. The Thompson sampler sampled the top-most common candidates from 100 calls to the predictive model posterior.

Off-the-shelf LLM models were replacing the acquisition functions. We tried Qwen3-32B (temperature 0.6), Llama-4-maverick-17b-128e-instruct-fp8 (temperature 0.7), and GPT-5 (medium reasoning effort). At each cycle, the LLM was prompted to select the optimal candidates to move on with from a table of candidates index, SMILES, prediction, and predictive model confidence. Prompts to models with more extended context windows (Llama-4-maverick) also included a table of all compounds used to train the predictive model and their respective labels (Prompt A.4.2). Models with smaller context windows (Qwen3, GPT-5) were informed of the accumulated observations by first having the model summarize the chemical space into a compact string (Prompt A.4.1), which replaced the historical data section. The set of candidates was then processed in chunks of $\sim$ 400 candidates for Qwen3 and GPT-5, and $\sim$1,200 candidates for Llama-4. The models were asked to select max(batch

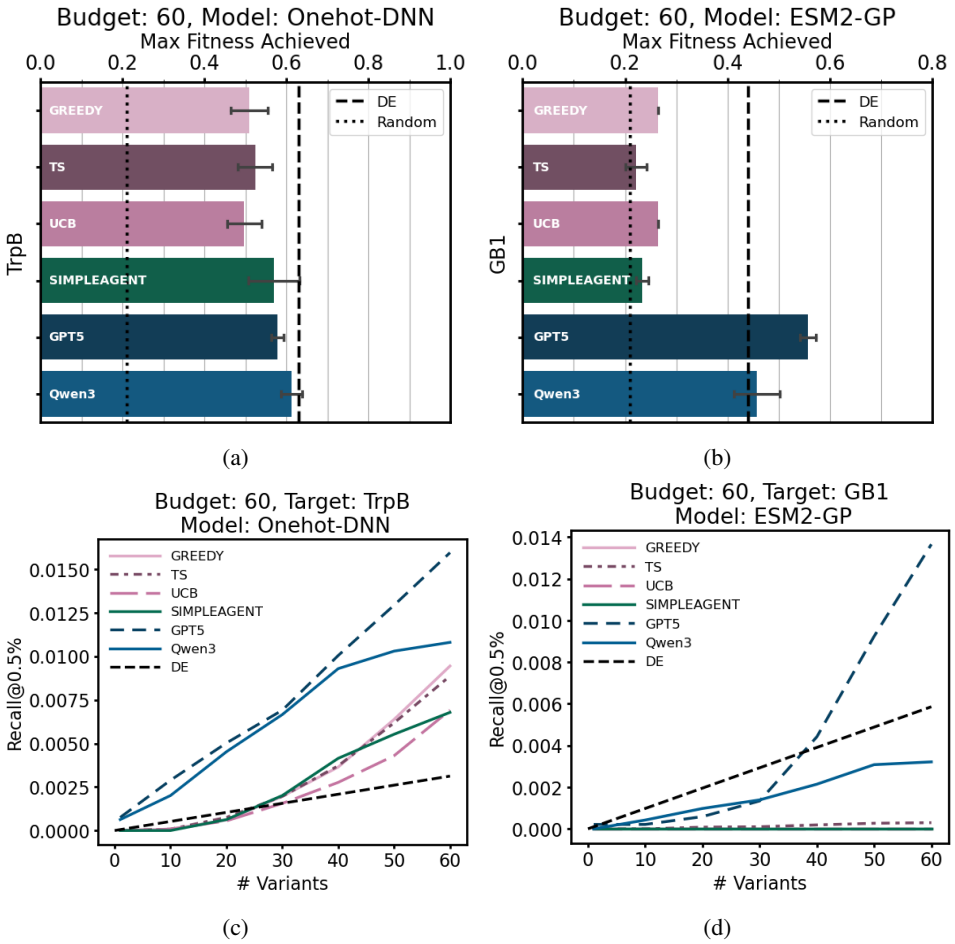


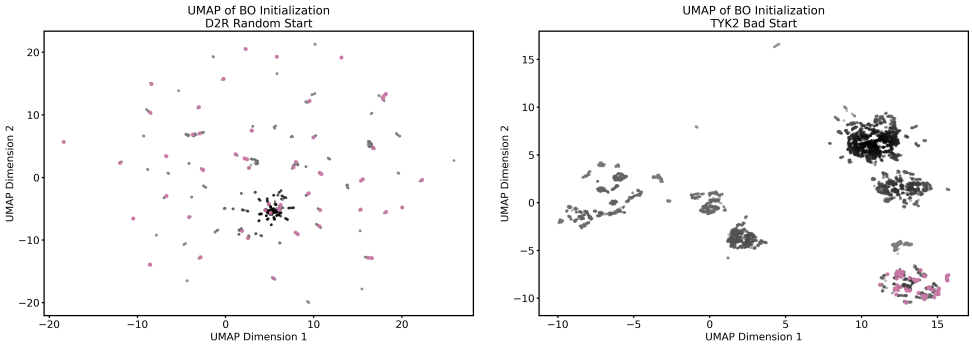
Figure 5: Complement to Figure 2. **Left column:** Max fitness and recall for campaign targeting a different protein motif (TrpB) with a budget of 60 and a batch size of 10 motifs. For TrpB, sequences closer to the WT generally yield higher fitness than for GB1 (Figure 2a), leading to DE outperforming all other methods in fitness and some in recall. Qwen3 primarily reproduces DE-like behavior, whereas GPT-5 explores more advanced patterns. **Right column:** Same campaign setup as Figure 2a, but with a poorly tuned predictive model. All methods relying on the predictive model fail, while reasoning models, being fully standalone, remain unaffected, demonstrating LLM robustness. Error bars denote standard deviation.

size, chunk size/n chunks) candidates from each chunk, and a final selection was then made from the chunk selections. Significant prompt engineering and fallbacks were required to make this method reliable and moderately efficient.

Agentic workflows were set up in Langgraph 0.5.0. They consist of three main LLM types operating in a hierarchical structure:

Strategist (Qwen3-32B, temperature 0.6): High-level planning node with reasoning enabled. Receives extensive information about the current BO stage, previous cycles, and accumulated observations (Prompt A.4.1). Generates an arbitrary number of complementary filter strategies for candidate selection. (`generate_strategy` in Figure 9). The only difference between the AGENT and the SIMPLEAGENT is a SMILES table (amino acid sequences in protein task) appended to the end of each cycle summary (cycle summary example, Prompt A.4.1).

Implementers (Claude Sonnet 3.5-20241022, temperature 0): Multiple execution nodes created in series, no thinking (`implement_strategy` in Figure 9). Each receives one strategy from the strategist and implements database filtering using prediction thresholds, UCB weighting, SMARTS matching parsed using SQL boolean format (AND, OR, NOT with parentheses) for flexible sub-



(a) TYK2, Random Start

(b) D2R, Bad Start

Figure 6: UMAP of the molecules selected as start batch (**Pink**) of all available molecules in the database (**Gray**) when initialized with a random start on D2R target (**Left**) and with a bad, clustered start on the TYK2 target (**Right**).

structure querying, and/or Tanimoto similarity to accumulated observations or the current batch. As implementers in the series select candidates, these are removed from the database to avoid selecting duplicates. If the batch size is not filled after processing all strategies due to incompatible filters, a final implementer node is created to fill the batch size with all strategies.

Summary node (Claude Sonnet 3.5-20241022, temperature 0): Final summarization node that processes all conversations from the implementation nodes, and generates comprehensive cycle selection reports including rationales and identified issues. This report is then used in the following cycle strategist prompt to inform it about successful/problematic filters.

The yellow-highlighted region in the workflow graph (Figure 9) represents the core implementation loop where strategies are executed through available tools.

To test how much the strategist was held back by the limited tools in the molecular domain, we removed all tool descriptions and stated "your strategies will be implemented by experienced chemists". The only filtering method commonly requested, not available to our agent was k-means clustering of the data.

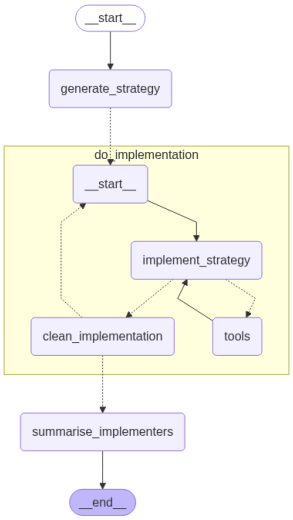


Figure 9: LLM workflow structure showing strategist-implementer-reporter hierarchy with tool integration loop.

A.2.4 Protein optimization task

We built on the codebase and data assembled by Yang et al. [2025]. The dataset consists of fitness scores of an almost complete set of possible mutations on four-residue motifs on two proteins (GB1 and TrpB). In the paper, they benchmark using the highest discovered fitness across four cycles (`batch size=96, starting size=96, budget=480`). We used DNN ensemble predictive model with one-hot encoding of the amino acids, optimized by Yang et al. [2025]. For comparison, we also test the implementation of the GP with ESM2 embedding, which was shown to perform the worst on the task. The acquisition functions Greedy, Thompson, and UCB ($\beta = 4$), were left identical to the paper. For more details, we refer to the original paper. We ran 4 campaigns: The **Small-Good 1**, and **Small-Good 2** campaigns assess the performance on short campaigns with optimized predictive model-encoding combination against two different protein targets (GB1, TrpB). **Small-Bad** evaluates the performance of acquisition functions when paired with an improper predictive model-encoding combination, whereas **Big-Good** assesses performance on the larger campaign from the original paper, utilizing an optimized predictive model and encoding. We evaluated performance by highest fitness, as in the paper, as recall with a 0.5% cutoff, which ensures both

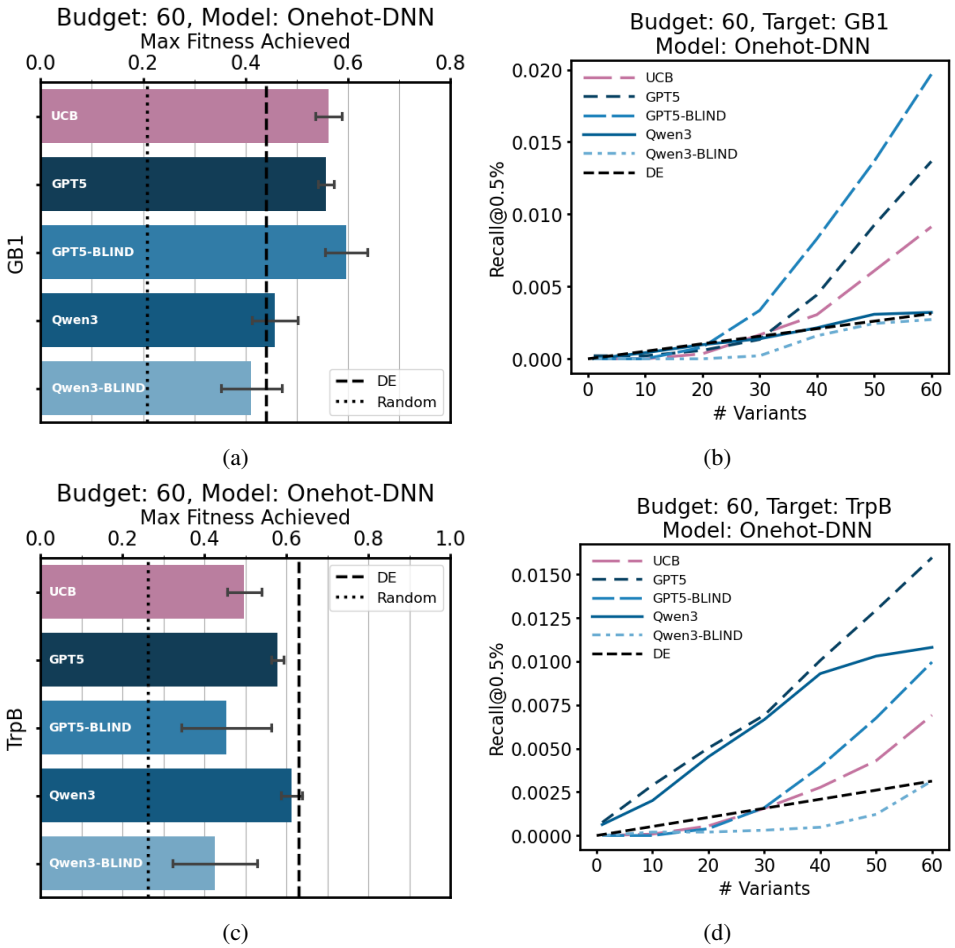


Figure 7: Performance of off-the-shelf models initialized without any context to the target protein (**-BLIND**) on the GB1 task (**Upper**), and the TrpB task (**Lower**), compared to a subset of other models. The GPT-5 blind model performs similar or significantly better than all statistical models on the recall metric. The unexpected performance on the GB1 task is largely a coincidence. GPT-5-BLIND often starts the search with the conservative guess "AAAA" which happens to be highly successful in the GB1 task. Qwen3 perfroms point wise mutations achieving performance similar to directed evolution (**DE**), which is an especially good strategy on the TrpB task, but fails to generalize to the GB1 task. Error bars mark standard deviation.

	Target	Model	Initial size	Batch size	Budget
Small-Good 1	GB1	DNN-onehot	10	10	60
Small-Bad	GB1	GP-ESM2	10	10	60
Big-Good	GB1	DNN-onehot	96	96	480
Small-Good 2	TrpB	DNN-onehot	10	10	60

Table 5: BO scenarios

WT are outside the cutoff. The 2 % cutoff used in Domain 1 quickly becomes saturated and less informative on this task.

Off-the-shelf models were implemented as generative models, completely standalone from the predictive models. The campaign was started of with an introduction to the task, the WT, and a brief description of the target motif function and protein (Prompt A.4.1). For example, the background for GB1 was: *"The target is a four-site epistatic region (wildtype: V39, D40, G41, V54, fitness 0.1) of the 56-residue protein G domain B1 (GB1), an immunoglobulin-binding domain from Streptococcal*

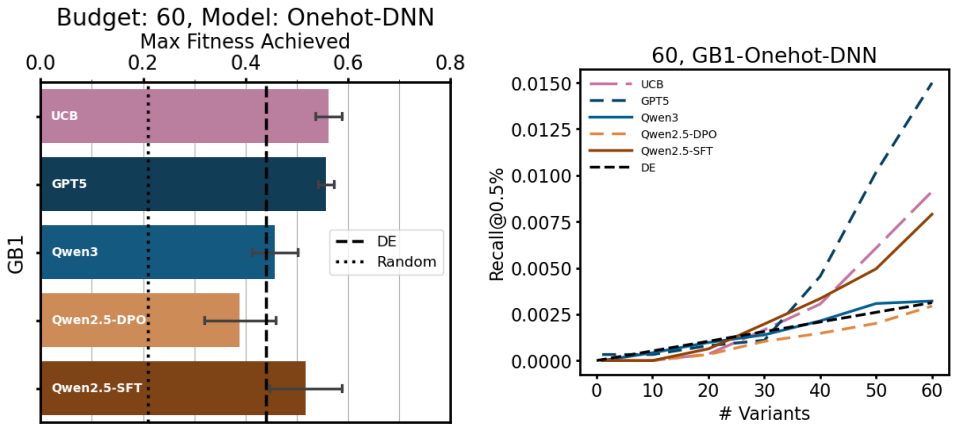


Figure 8: **GB1**. SFT and DPO models compared to a subset of other methods. The SFT model significantly outperforms the reasoning model Qwen3 while executing 30x faster and achieves performance competitive with UCB on the recall metric Error bars mark standard deviatio.

bacteria. These sites account for a majority of the most strongly epistatic interactions in GB1 and span a fitness landscape of 160,000 variants. Variants were assessed for IgG-Fc binding using mRNA display and high-throughput sequencing". Importantly, the actual protein name "TrpB" was changed to "protein" as we observed a significant bias in Qwen3 to the amino acid Tryptophan (W,Trp) introduced by simply including the word TrpB in the prompt, completely deteriorating performance. Indeed, Trp occur 584x less frequently in the top 0.5 % performing sequences compared to average.

After each generation, it was again prompted with *The validation experiment in cycle {current_cycle} is finished. These are the results: {validated_results}.* and a system message reminding it about the output format, and the 10 best sequences found so far (Prompt A.4.1). The model was asked to generate 50 % overhead to each batch selection and rank the sequences in order of importance to avoid issues with overlapping generations, and instances of generations not existing in the labeled data. If too many sequences were invalid, the model was prompted to correct mistakes, or informed that "the following were invalid for experiments" and asked to generate new. To manage context limits and gracefully handle formatting issues, the off-the-shelf reasoning models were wrapped in a Langgraph context. This context allowed nodes to validate outputs, ask the model to correct errors, summarize conversations during longer campaigns, and extract sequences from prompts (Figure 10). Blind models were implemented with the same prompts, but replacing the background with an empty string.

The LLM workflows were implemented identically to the workflows in the molecular domain, but with different tools. The models were able to sort by Hamming and Blosum62 similarity, regex, prediction, and UCB. Just as in the molecular task, we implemented an AGENT informed by the accumulated observations in the form or a table of sequences and fitness, and a SIMPLEAGENT without detailed data. The AGENT was only used for one task as we found SIMPLEAGENT greatly outperformed it. We also found that removing information about the predictive-model prediction range, fitness ranges, and letting the LLM rewrite tool descriptions improved performance and consistency further.

A.2.5 Fine-Tuning

We aimed to determine if a non-reasoning LLM could be fine-tuned to perform a generative task. Wang et al. [2025] showed that training a non-reasoning LLM on acquisitions by a statistical model in an artificial setting can improve the Bayesian behavior of the LLM. We created 14 biologically relevant synthetic datasets from the ESM2 embedding of the motifs provided by Yang et al. [2025]. Fitness was assigned to the sequences using Algorithm 1, which is able to generate a large variety of fitness distributions. Step 9-13 are optional and were taken to increase the difficulty. No parameters were tuned significantly and all were assigned to create variance in the distributions while keeping them somewhat similar to the biological data. We ran 50 campaigns (`batch size=10, initial size=10, budget=400` on each dataset using onehot embedding, DNN ensemble predictive model

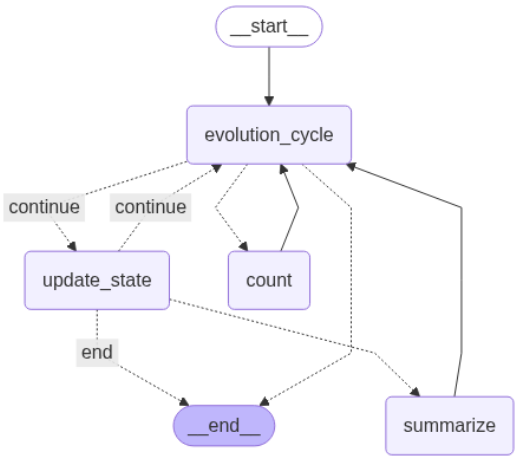


Figure 10: LLM standalone BO structure showing the **evolution node** which prompts the LLM to analyze and generate new variations for the next cycle, the **count node** which gives the LLM feedback if generations are duplicates or invalid, the **update state node** which labels the proposed variations, and a simple **summary node** which triggers if the message stream approaches context limit. The loop ends after a set number of cycles.

and TS acquisition function. From each trajectory, we drew 15 time points t with $P(X = t) = t^{-1} \sum_{i=1}^{40} i^{-1}, t \in [1, 40]$. For each time point, we created a prompt where the past trajectory selections were added in context with their assigned fitness, and the next batch selection in the trajectory was used as a ground truth optimal selection. To further increase variance between the prompts, we permuted all sequence positions in each prompt and shuffled the order of the sequences in-context. This step was crucial for successful training. We attempted both Supervised Fine-Tuning (SFT) with the ground truth as a label, and Direct Preference Optimization (DPO) [Rafailov et al., 2024] with the ground truth contrasted with a randomly generated string or a string in the set of accumulated observations.

SFT was run using SFTTrainer in TRL (trl 0.19.1, transformers 4.54.1, flash-attn 2.7.4, vllm 0.10.0) [von Werra et al., 2020] with `per_device_batch_size=8`, `gradient_accumulation_steps=2`, `dtype=bf16`, `gradient_checkpointing=True`, `max_grad_norm=1`, `weight_decay=0.1`, `learning_rate=5e-6`, `warmup_ratio=0.05`, `lr_schedular_type=cosine` for 3 epochs, and with early stopping on evaluation loss. Training was only made on the responses and not prompts. DPO was run with `per_device_batch_size=1`, `gradient_accumulation_steps=16`, `dtype=bf16`, `gradient_checkpointing=True`, `max_grad_norm=1`, `weight_decay=0.1`, `learning_rate=5e-7`, `beta=0.9`, `lr_schedular_type=cosine` for 2 epochs. The strong regularization to the reference model was essential for any results. DPO has been shown to work well for training LLMs to think like statistical models, but for this task, it performs poorly. The reason is likely that the difference between the chosen and rejected strings is too similar. The difference between an intelligently mutated string that deviates from most high performers by up to 2 positions, and a string generated at random can be almost identical. First training the model with SFT and then further with DPO reduces performance compared to only training with SFT.

Qwen2.5-7B-Instruct was loaded from HuggingFace. The off-the-shelf model underperformed random selection because of formatting issues and an inherently greedy behavior. Training on 4 NVIDIA A100-SXM4-80GB with accelerate and Deepspeed [Rasley et al., 2020] takes about 12 min for 0.5B models and 2 hours for 7B models.

The fine-tuned model was optimized to generate 10 sequences per batch and was run with a temperature of 0.7 during inference. When the model generated an invalid sequence, the in-context data was reshuffled and the model re-prompted. When the model generated duplicates of the in-context data, temperature was incremented by 0.05 in that cycle. Running a small campaign (`batch size=10, starting size=10, budget=60`) takes 11.9 s without any time optimization compared to 15.6 s with onehot+GP+Thompson (Proper posterior sampling of posterior approximated with 1000 Fourier features) on a single NVIDIA A100-SXM4-40GB.

Algorithm 1 Synthetic Fitness Generation

Input: Sequences $\mathcal{S}$
Tokenize: $\mathbf{E} \leftarrow \mathbf{ESM2}(\mathcal{S})$,$\mathbf{E} \in \mathbb{R}^{n \times d}$
Sample binary mask: $\mathbf{m} \sim \text{Bernoulli}(p = 0.2)^d$
Sample log-weights: $\log \mathbf{w} \sim \mathcal{N}(\mu = -6.5,\ \sigma^2 = 1.5)$
Apply mask: $\mathbf{w} \leftarrow \mathbf{m} \odot \exp(\log \mathbf{w})$
Compute fitness: $\mathbf{f} \leftarrow \mathbf{E} \cdot \mathbf{w}$
Add noise: $\log \mathbf{f} \leftarrow \mathcal{N}(\log \mathbf{f},\ 0.01^2)$
Normalize: $\mathbf{f} \leftarrow (\mathbf{f} - \min f)/(\max f - \min f)$
Set 10% randomly to zero: $f_i \leftarrow 0$ for $i \in \mathcal{I} \subset \{1, \ldots, n\}$
Set near-zero values to zero: $f_i \leftarrow 0$ if $f_i < 0.001$
while $\text{Quantile}_{0.995}(\mathbf{f}) > 0.5$ **do**
    $\mathbf{f}_i \leftarrow \mathbf{f}_i^{1.5}$
end while
return Dataset $\{(s_i, f_i)\}_{i=1}^n$

A.3 Statistical Analysis

Statistical intervals were defined as margin of error for a 95% confidence in population mean, defined as $e = \frac{\sigma}{\sqrt{n}} t_{0.975,n-1}$.

A.3.1 Bootstrap

Significance threshold was defined as 95% confidence in separation between a statistical method and all LLM-based methods, or an LLM-based method and all statistical methods. When comparing agents, or off-the-shelf models in the molecular task, to statistical methods, bootstrapping was made on shared random seeds ($N = 10$ for agent-statistical comparisons, $N = 5$ for Qwen3-statistical comparisons, $N = 3$ for GPT-5-statistical comparisons) because of the strong covariance between trajectories with the same starting point. N random seeds were drawn with replacement, the mean of the corresponding trajectories were compared between methods.

When comparing statistical methods to off-the-shelf models in the protein task, bootstrapping was made on trajectories as the off-the-shelf models here are completely starting-point independent. We could therefore use a higher sample size for the statistical models. $N = 50$ random seeds and their respective trajectories were drawn with replacement from the set of random seeds used to initialize the statistical models, and $N = 10$ trajectories were drawn with replacement from the set of off-the-shelf LLM trajectories. The mean of the corresponding groups of trajectories were compared between methods.

All bootstrap iterations were repeated 10,000 times.

A.4 Prompts and Responses

A.4.1 Prompts

Prompt 1: In-context acquisition prompt. "Oracle" here refers to the predictive model

```
You are selecting ligands for validation in an active learning campaign for
    protein {protein}.

**OVERALL CAMPAIGN OBJECTIVE:** Maximize the total number of top high-
    affinity ligands in the training data at the end of the campaign. A top
    high-affinity ligand is in the top 2% of all ligand candidates.

**Campaign Status:**
- Cycle: {cycle}/{total_cycles}

**Historical Data:**
Used to train the Gaussian Process Regression oracle, sorted by RBFE:
<validated_ligands>
{self._compact_df(labeled_data, index=False)}
</validated_ligands>

3. **Candidates** [SMILES, predicted RBFE, std]:
Randomly ordered.
<candidates>
{self._compact_df(chunk)}
</candidates>

**Your task:**
- Reason about the chemical space, the candidates, the reliability of the
    oracle model, and the overall campaign goal. Describe the rationale
    behind your selection.
- Select exactly {budget} candidates to move on with (validate and add to
    training data for next cycle). Output format:
<selected_indices>
[index1, index2, ...]
</selected_indices>
```

Prompt 2: Prompt to summarize chemical space

```
You are a chemoinformatic expert working in a team of chemoinformatic experts
     selecting ligands for validation in an active learning campaign protein
    {protein}.
You have access to a set of validated ligands, and your junior colleague has
    access to a set of thousands of candidates to validate.
**Your task:**
- Create a concise summary of the validated chemical space that preserves all
     structural features, patterns, and structure-activity relationships,
- Your junior colleague will use the summary to select candidates from his
    large list. Include any crucial instructions for the selection process.
    Your junior collegue will not have access to the training data.
- Consider number of validated ligands, and number of cycles left. What are
    the most important features to preserve?

**Campaign Status:**
- Current cycle: {cycle}
- Total cycles: {total_cycles}
- Oracle model: {oracle_name}
- Overall Campaign Goal: Maximize number of validated high-affinity ligands
    in the training data at the end of the campaign
```

```
**Validated Training Data:**
{len(training_df)} ligands with measured RBFE values:
<validated_ligands>
{self._compact_df(training_df)}
</validated_ligands>

**Required output:**
<training_data_description>
Comprehensive but concise description of the training data to be used by the
    junior colleague that fully encompasses the chemical space, and the
    active learning context.
</training_data_description>

**Output Format:**
<training_data_description>
[Training data description that will be passed to the junior colleague]
</training_data_description>
Include exactly one instance of <training_data_description> tag in your
    response
```

Response 2: Example of Qwen3 summary of chemical space

```
<training_data_description>
The validated ligand set (120 molecules) centers on **bipartite scaffolds**
    linking aromatic/heteroaromatic rings via **amide/urea linkers**, with
    key features:
1. **Core Architecture**: Two aromatic rings (often phenyl/pyridine) bridged
    by amide/urea (-CONH-/-NHC(O)N-) linkers. Common substructures include *
    ccnc(c)*âĂŞ*C(=O)N*âĂŞ*cccccX* (X = Cl/F/Br/O).
2. **Electron-Withdrawing Groups**: Para/meta halogens (Cl, F, Br) dominate
    on terminal aromatics (e.g., *c(c(Cl)F)*, *c(c(Cl)Cl)*). Dual
    fluorination (*c(c(F)F)*) improves RBFE (>9.5 kcal/mol).
3. **Hydrogen Bonding Motifs**: Urea/urethane linkers (*N=C(O)N*) enhance
    activity (top 10% RBFE âĹě9.5 kcal/mol). Terminal nitroso/amino groups
    (*-N(H)C=O*, *-NH2*) reduce activity (<8.5 kcal/mol).
4. **Aliphatic Chains**: N-linked methyl/ethyl groups (*-CH3*, *-CH2CH2-* in
    *N(C)C(=O)N*) stabilize topologies. Longer chains (*-CH2CH2CH2-*)
    correlate with reduced affinity (ÎŤRBFE âĹĹ -0.3âĂŞ0.5 kcal/mol per CH2).
5. **Steric Constraints**: Bulky substituents (e.g., *Br*, *N(C)(C)C*)
    tolerate modestly (RBFE >9 kcal/mol if paired with strong ÏĂ-stacking
    partners); avoid *I* or *NO2* due to poor performance (median RBFE ~7
    kcal/mol).
6. **Orthogonal Features**: Ether linkers (*-O-CH2-*) improve water
    solubility without sacrificing activity (best performers retain Cl/F on
    aromatic rings).

**Selection Priorities for Next Cycles**:
- Fill gaps in **meta-fluorinated aromatics** and **mixed X/Y halogen pairs**
     (e.g., *c(c(Cl)F)* vs *c(c(F)F)*).
- Test **shortened/cyclic spacers** (replace *CH2CH2* with *CH2* or *CH2OCH2
    *) to optimize linker stiffness.
- Explore **triazine/triazole cores** as rigid alternatives to *ccnc(c)*
    scaffolds.
- Avoid redundant modifications (e.g., additional Br/I substitutions or
    longer aliphatic chains).
- Validate **heteroaryl replacements** (thienyl, furyl) adjacent to amide
    linkers.
```

```
Candidates must align with existing SAR while introducing minimal scaffold
    novelty. Favor analogs expanding **fluorination patterns** and **
    alkylation balance** across the bipartite framework.
</training_data_description>
```

Response 3: Example of GPT5 summary of chemical space

```
<training_data_description>
Context and goal
- You will down-select candidates (no access to training data) for TYK2
    across 2 remaining cycles (currently 7/9). Oracle = Gaussian Process
    regressor (uncertainty-aware). Objective = maximize the number of
    validated high-affinity ligands in the training set by end of cycle 9.

What has worked (validated chemical space; 418 ligands; top RBFE up to ~12.2)
- Two privileged chemotypes dominate. Keep these intact unless explicitly
    exploring:
  1) "Triad" anilide series (majority, highest RBFE):
     Left: 6-membered diazine/pyridine core, most often a 4-aminopyrimidine/
    pyrimidinyl-pyridine, typically 5-alkyl (Me >> Et ~= iPr) and para-amine
    bearing a small basic side chain (see "Left-side chains" below).
     Middle: 2-aminopyridine (or 2-aminopyridinyl) most often 5-fluoro-
    substituted.
     Linker: secondary anilide/benzamide (--NH--C(=O)--Ar) with the amide NH
    on the "middle ring" side (do not reverse the amide).
     Right (distal aryl): dihalo phenyl; the recurring best pattern is ortho-
    chloro + meta/para-fluoro; di-F is also strong; o-Cl/o-Cl is acceptable;
    o-Br variants can be good. Occasional phenol tolerated but generally not
    top.
     Summary motif (abstract): [5-Me-(amino)pyrimidine/pyridyl]--NH--(5-F-2-
    aminopyridyl)--NH--C(=O)--[o-Cl, m/p-F phenyl].
  2) "Morpholine-tail heteroaryl amide" series (secondary cluster; many
    9.8--11.7):
     Acyl aryl similar to above (often F/Cl patterns).
     Hinge-facing heteroaryl is more N-rich (e.g., Nc--cnn(c)--) bearing a
    pendant N-morpholine or N-(2-oxa-5-azabicyclic) tertiary amine (--
    N4CCOCC4). Keep the tertiary amine and the heteroaryl arrangement
    together; they are synergistic.

Left-side chains (key to high RBFE, in order of priority)
- Small, conformationally constrained cations:
  A) Azetidine (--N1CCC1--) on the left ring amine: repeatedly among the top
    (~11--12.2).
  B) Cyclopropyl-bearing secondary amines (--NC1CC1--) and small hydroxyalkyl
     secondaries (--N--CH(CH3)--CH2OH or --N--CH2--CH2OH) are strong.
  C) 4-hydroxyazetidine and other compact "N,O" motifs are good compromises.
  D) Bulkier or more flexible amines (long chains, multiple heteroatoms) and
    heavily N-methylated cations tend to underperform vs A--C.

Substituent SAR you can apply directly
- Distal aryl (acyl side):
  Priority 1: o-Cl + m- or p-F (best-in-class recurring motif).
  Priority 2: o-Cl + p-Cl or o-Cl + p-Br (slightly lower on average, but
    still strong).
  Priority 3: di-F (good); di-Cl (acceptable; generally slightly lower).
  Deprioritize: strong donors (p-OMe/phenoxy), dense polar patterns, or
    removal of ortho-halogen.
- Middle ring:
  2-aminopyridine with 5-F consistently outperforms unsubstituted analogs;
    removal of the 5-F is generally a drop.
```

```
- Left heteroaryl core:
  4-aminopyrimidine/related diazines with a 5-Me substituent are most common
    in top tier. Me > Et ~= iPr; larger alkyls drop off.
- Amide orientation:
  Preserve the "middle ring --NH--C(=O)-- distal aryl" orientation. Amide
    reversal/urea/sulfonamide replacements are not supported by top data.
- Halogen count:
  Retain at least one ortho-halogen on the distal aryl (Cl >> F for ortho);
    removing it usually costs 0.5--1.0 RBFE.
- H-bond pattern:
  Keep exactly one amide NH donor and two ring nitrogens across the triad;
    adding extra strong donors/acceptors generally hurts.

What to avoid (seen repeatedly lower)
- Reversing the amide, swapping the amide for urea/sulfonamide, or breaking
    the triad topology.
- Overly flexible or bulky cationic tails; long aliphatic chains; multiple
    extra heteroatoms on the left side chain.
- Distal aryl without ortho-halogen; heavy electron-donation (e.g., para-OMe)
     unless paired with the optimal ortho-halogen pattern.
- Extra ring nitrogens that overpolarize the middle ring (e.g., turning the 2-
    aminopyridine into a more N-rich diazine) typically drops RBFE.
- Nitrile decorations and phenols appear in mid-tier but rarely at the top;
    deprioritize unless needed for diversity.

Concrete selection rules for your candidate list
1) Hard filters (must pass)
   - Contains the triad or morpholine chemotype:
     a) Triad: left 6-membered N-containing ring (prefer 4-aminopyrimidine)
    linked to 2-aminopyridine (prefer 5-F), linked via --NH--C(=O)-- to a
    distal aryl.
     b) Morpholine series: aryl amide paired with Nc--cnn(c)--(N-morpholine)
    heteroaryl tail.
   - Distal aryl includes an ortho-halogen (prefer Cl; Br acceptable; F only
    if paired with another F/Cl).
   - Exactly one amide linker (secondary amide; not reversed).
2) Prioritization scoring (apply additively; pick highest-scoring per cluster
    )
   +3: Left side chain is azetidine (--N1CCC1--).
   +2: Left side chain is cyclopropyl secondary amine or small hydroxyalkyl
    secondary amine.
   +2: Middle ring = 2-aminopyridine with 5-F.
   +2: Distal aryl = o-Cl + p-F (or o-Cl + m-F).
   +1: Distal aryl = o-Cl + p-Cl (or o-Cl + p-Br) or di-F.
   +1: Left ring = 4-aminopyrimidine with 5-Me.
   -2: Reversed amide, missing ortho-halogen, or bulky/flexible tertiary
    amines on the left.
   -1: Extra strong donors (phenols, guanidines) or >1 extra heteroatom
    beyond the patterns above.
3) Diversity protocol (to hedge with GP uncertainty; 2 cycles left)
   - Cycle 7->8 (this round): 70% exploitation + 30% exploration
     Exploitation: Pick close analogs within the triad keeping the features
    above; systematically vary ONE position at a time:
       R-distal: swap p-F <-> p-Cl <-> p-Br; move F m<->p; di-F vs o-Cl/p-F.
       R-middle: 5-F -> 5-H (limited set) to quantify the drop; 2-
    aminopyridine <-> 2-aminopyrimidine (very small set).
       R-left chain: azetidine <-> hydroxyethyl secondary amine <->
    cyclopropyl; Âśsingle N-methyl; Âśone OH.
     Exploration: 3--4 small clusters outside the densest neighborhood:
       - Replace distal phenyl with 2-pyridyl retaining o-Cl (on 3-position
    relative to amide) and a para-F where possible.
       - Swap 5-Me on the left ring to Et or iPr (single change) to probe
    steric tolerance.
```

```
      - Morpholine-series pickups that match the same distal aryl patterns (
    o-Cl with F).
   - Cycle 8->9 (final): 85--90% exploitation
     - Lock in the top 2--3 R-group patterns discovered; make nearest-
    neighbor analogs (1 change only; keep ortho-Cl and 5-F).
     - Reserve 10--15% for "safety" variants: same distal aryl but two best
    left-chain options; same left chain but two best distal patterns.

Quick triage checklist (yes/no)
- Amide present and correctly oriented? Yes -> proceed; No -> drop.
- Distal aryl has ortho-Cl (or Br)? Yes -> proceed; No -> only keep if di-F
    and everything else optimal.
- Middle ring is 2-aminopyridine with 5-F? Yes -> strong; No -> keep a few
    for diversity only.
- Left chain is compact cation (azetidine/cyclopropyl/hydroxyethyl)? Yes ->
    strong; No -> deprioritize.
- Left heteroaryl is 4-aminopyrimidine (5-Me)? Yes -> strong; No -> keep
    minority for exploration.

If you must choose one "safe" pattern to maximize hits now
- Pick candidates matching: 4-aminopyrimidine (5-Me) --NH-- 5-F-2-
    aminopyridine --NH--C(=O)-- 2,4-dihalo phenyl (o-Cl + p-F), with a left
    azetidine side chain. Then explore Âś(p-F->p-Cl), Âś(azetidine->
    cyclopropyl/hydroxyethyl), Âś(o-Cl/p-F->o-Cl/m-F).

What not to change this late
- Do not remove the ortho-halogen on the distal aryl or the 5-F on the middle
     ring across the bulk of your picks.
- Do not invert the amide or replace it with urea/sulfonamide.
- Do not add bulky solubilizing tails; prefer small, rigid cations.

Submission format suggestion (to help us review quickly)
- Group selected candidates into small R-group matrices:
  Matrix 1 (triad, fixed left = azetidine; vary distal p-substituent): {o-Cl/
    p-F, o-Cl/p-Cl, o-Cl/p-Br, o-Cl/m-F}.
  Matrix 2 (triad, fixed distal = o-Cl/p-F; vary left chain): {azetidine,
    cyclopropyl secondary amine, hydroxyethyl secondary amine, 4-
    hydroxyazetidine}.
  Matrix 3 (morpholine series, fixed distal = o-Cl/p-F; vary heteroaryl N
    pattern minimally).
- Within each matrix, avoid >1 simultaneous change relative to the
    prototypical pattern.

Rationale link to RBFE observations
- The highest RBFE ligands co-occur with: compact left cations (especially
    azetidine), 5-F on the central 2-aminopyridine, and ortho-Cl plus F on
    the distal phenyl; Br is tolerated. When these features are removed,
    drops of ~0.3--1.0 RBFE are common; when combined, RBFE >=10.5--12 is
    frequent. The morpholine-tail chemotype is a reliable secondary path with
     the same distal aryl patterns.

Use this guide to pre-filter and prioritize; then, if you have a model to
    score uncertainty, allocate the exploration quota to the edges defined
    above.
</training_data_description>
```

Prompt 3: Strategist prompt. "Oracle" here refers to the predictive model

```
You are a senior medicinal chemist and expert in active learning performing
    an active learning drug development campaign. Your task is to evaluate
```

```
    the current state of the campaign and delegate candidate selection to
    junior colleagues who have strong computational skills but limited
    chemical intuition. They need clear, specific instructions to filter the
    molecular database effectively.

**CONTEXT AND DATA EXPLANATION:**
- SMILES: Chemical structure representation of molecules
- affinity: Validated binding affinity from experiments (RBFE score, higher
    is better)
- predictions: Oracle model's predicted affinity for unlabeled candidates
- std: Oracle model's standard deviation in predictions
- max_per_cycle: Maximum affinity achieved in each cycle's selected compounds
- mean_per_cycle: Average affinity of selected compounds in each cycle
- oracle RMSE: Root Mean Square Error between oracle predictions and
    validated affinities for selected compounds at a given cycle (measurement
     of the oracle's performance on selected compounds, not nessecairily the
    oracle's performance on the entire database. If we heavily favor
    exploitation one cycle, RMSE will appear lower because we are selecting
    candidates that the model is confident about, but not necessarily the
    best candidates.)

- Completed cycles: {cycles_completed}
- Total cycles: {total_cycles}
- Budget this cycle: {batch_size}
- Oracle model: Gaussian Process Regression
- Measure of affinity: RBFE (higher is better)
**Overall Campaign Goal**
From our ligand database, discover as many compounds as possible that would
    rank in the top 2% by binding affinity. Success is measured by recall: of
     all the truly high-affinity ligands in the database, how many can you
    find within your experimental budget?

**Previous cycles performance**
- Max affinity in selected candidates (starting from cycle 0, should increase
     over time): {max_per_cycle}
- Mean performance in selected candidates (starting from cycle 0, should
    increase over time): {mean_per_cycle}
- Oracle RMSE on selected candidates a given cycle (starting from cycle 0,
    not included in training data that cycle, should decrease over time): {
    oracle_rmse_per_cycle}


**Previous Cycles Summaries:**
All Validated Candidates Used for Training or Current Cycle Oracle. Here
    follows strategies implemented at previous cycles, and their respective
    performance. Use this to inform your new strategies in the context of the
     current cycle and campaign state, dont copy them.
<past_cycles_data>
{past_cycles_data}
</past_cycles_data>

**ANALYSIS:**
Analyze the current state of the campaign.

**CHEMICAL ANALYSIS REQUIREMENTS:**
Identify chemical regions for exploitation and exploration:
- Be specific: use chemical knowledge and terminology to describe patterns
    and potential binding motifs. Look for both simple and complex (multiple
    substructures) patterns.
- Similar targets: [Structural patterns in the training data that are
    promising]
- Potential targets: [knowledge about protein {protein} that could be used to
     guide the selection, hypothesis testing motifs]
```

```
- Coverage gap: "Training data covers a [small/medium/large] portion of the
    relevant feature space"
- What targets were explored in previous cycles? How did they perform?

**Active Learning Status:**
Identify if exploration or exploitation is more important:
- How are we doing towards the overall campaign goal?
- Passed progress: [is the current training data promising or is the oracle
    plateauing? Are we stuck in a local optimum?]
- Exploration vs exploitation: [can we afford to explore more or should we
    exploit more?]
- Is the oracle better or worse that our intuition?

**TASK:**
Design the optimal selection strategy for this cycle that your junior
    colleagues can execute independently. Your junior colleagues don't know
    about each other's work. This may be a single protocol or multiple
    complementary protocols, depending on what's most appropriate for the
    current campaign state. Design hypotheses for testing and informing
    coming cycles when suitable.
Provide only actionable filtering instructions - no explanations or chemical
    rationale needed.

YOUR JUNIOR COLLEAGUES' CAPABILITIES
They can filter candidates using:
- Computational approaches: Predictions
- Upper confidence bound (UCB): Predictions + beta * std, given beta
- Chemical approaches: Substructures (SMARTS or substructure names) and
    similarity metrics
- Diversity approaches: Tanimoto similarity metrics (pairwise between
    selected candidates or to training data)
- Hybrid approaches: Combining the above

EACH PROTOCOL SHOULD SPECIFY:
Exact number of candidates to select
Precise filtering criteria with numerical thresholds
Clear chemical and/or computational constraints

Ensure total candidate count across all protocols equals {batch_size}. Output
     your analysis and strategies as soon as you are confindent in your
    selection.
```

Response 4: Example of cycle summary. "Oracle" here refers to the predictive model

```
Cycle 1:
# AL Campaign Acquisition Summary

## Implementation
All selection strategies were successfully implemented, yielding 60 total
    compounds:
- 25 compounds balancing exploitation/exploration (pred >7.5, UCB beta=1.5)
- 20 exploration-focused compounds (UCB beta=3.0, high uncertainty)
- 10 structurally diverse compounds (Tanimoto <0.3)
- 5 compounds exploiting known SAR (fluorinated pyrimidines with halogens)

No substructure filters failed or needed to be released. Database contained
    all requested chemical patterns.

## Campaign Impact
```

```
The selections maintain a strategic 42/58 split between exploitation and
    exploration:
- Exploitation (30 compounds): Targeting high-affinity regions through
    predicted values >7.5 and known SAR patterns of fluorinated pyrimidines
- Exploration (30 compounds): Probing uncertain regions through high UCB
    scores and ensuring structural diversity

This balanced approach is appropriate for the early campaign stage with
    limited training data (60 compounds). The selections will expand chemical
     space coverage while still leveraging emerging SAR patterns around
    halogenated aromatics. The varying uncertainty thresholds (beta=1.5 vs
    beta=3.0) provide a gradient of exploration intensity, helping to
    systematically improve model reliability.

Strategy 1: Select 25 candidates with predictions >7.5 and UCB (beta=1.5) to
    balance high-affinity exploitation with moderate uncertainty
Number final selections: 25
RMSE: 0.76
Mean: 9.60
Max: 10.63

Strategy 2: Select 20 candidates with UCB (beta=3.0) to prioritize
    exploration of uncertain regions (std >0.45)
Number final selections: 20
RMSE: 1.14
Mean: 8.92
Max: 10.87

Strategy 3: Select 10 candidates with Tanimoto similarity <0.3 to the
    training set to enforce diversity
Number final selections: 10
RMSE: 1.56
Mean: 8.29
Max: 9.57

Strategy 4: Select 5 candidates containing fluorinated pyrimidine core [n]1
    cccc([F])c1 and [Cl] or [Br] substitutions
Number final selections: 5
RMSE: 0.75
Mean: 10.04
Max: 10.33
Selected candidates from cycle 1 with oracle predictions and std:
SMILES|affinity|oracle_prediction|oracle_std
c1cc(c(cc1N)C(=O)Nc2cc(ncc2F)NC(=O)C3CC3)Cl|8.96|9.84|0.173
CNC(=O)Nc1cc(ccn1)NC(=O)c2c(cccc2Cl)F|9.95|9.68|0.252
c1cc(c(cc1N)Cl)C(=O)Nc2cc(ncc2F)NC(=O)C3CC3F|10.5|9.7|0.178
...
```

Prompt 4: Standalone Model Start Prompt

```
You are an expert protein engineer with deep chemical intuition leading a
    directed evolution campaign. Apply rigorous chemical principles to
    discover high-fitness variants within your experimental budget.

**CAMPAIGN OVERVIEW:**
- Target: Four-site region of {self.protein}
- Background: {background}
- Total Budget: {self.total_budget} experimental validations
- Batch Size: {self.batch_size} sequences per round
- Number of cycles: {self.max_cycles}
```

```
- Sequence Length: 4 amino acids

This is the START of your campaign. You have no prior data.

**STRATEGIC APPROACH:**
At every cycle, you'll receive previous results. Select {3*self.batch_size //
     2} untested sequences using chemical logic and hypothesis testing.

**OUTPUT REQUIREMENTS:**
1. **Chemical Reasoning**: Evaluate the past cycles and explain your
    mechanistic hypotheses and chemical logic for the next cycle. Add
    reasoning within <reasoning> tags.
3. **Priority Ranking**: Sort by priority with chemical justification
4. **Final List**: End with {3*self.batch_size//2} ranked sequences for next
    cycle validation for buffer, {self.batch_size} of which will be validated
     in the next cycle.

Design sequences that test well-reasoned chemical hypotheses. Balance
    thorough exploration with chemical principles to maximize discovery
    potential through systematic, chemistry-guided experimentation.
```

Prompt 5: Standalone Model System Message

```
You are an expert protein engineer with designing and controlling a directed
    evolution campaign. You must strategically select protein sequences to
    maximize fitness discovery within your experimental budget at the end of
    the campaign.

Current cycle {state["current_cycle"]} of {state["max_cycles"]}.
Select {3*self.batch_size//2} sequences, optimal for the campaign goal, to
    validate in next cycle.
Of these, {self.batch_size} will be validated in the next cycle, depending on
     experimental feasibility.
Sort you selection by priority.

Highest performing sequences so far:
{self.validated_results.sort_values(by='fitness', ascending=False).head(10).
    round(3).to_string(index=False)}

**Output format**: End your analysis with a list of {3*self.batch_size//2}
    sequences (STRICTLY 4-letter codes like 'AMHG', 'QPEI').
```

A.4.2 Response examples

Prompt 6: GPT5 TrpB BLIND full campaign example.

```
You are an expert protein engineer with deep chemical intuition leading a
    directed evolution campaign. Apply rigorous chemical principles to
    discover high-fitness variants within your experimental budget.

**CAMPAIGN OVERVIEW:**-
- Target: Four-site region of unknown
- Background: No background available, we enter the campaign blind.
- Total Budget: 60 experimental validations
- Batch Size: 10 sequences per round
- Number of cycles: 6
- Sequence Length: 4 amino acids
```

```
This is the START of your campaign. You have no prior data.

**STRATEGIC APPROACH:**
At every cycle, you'll receive previous results. Select 15 untested sequences
     using chemical logic and hypothesis testing.

**OUTPUT REQUIREMENTS:**
1. **Chemical Reasoning**: Evaluate the past cycles and explain your
    mechanistic hypotheses and chemical logic for the next cycle.
3. **Priority Ranking**: Sort by priority with chemical justification
4. **Final List**: End with 15 ranked sequences for next cycle validation for
     buffer, 10 of which will be validated in the next cycle.
4. **Output format**: List 15 sequences (STRICTLY 4-letter codes like 'AMHG',
     'QPEI') at the end of your response.

Design sequences that test well-reasoned chemical hypotheses. Balance
    thorough exploration with chemical principles to maximize discovery
    potential through systematic, chemistry-guided experimentation.
```

Response 5: GPT5 response

```
Chemical reasoning and strategy for Cycle 1 (blind start)
- With no prior data, we should maximize information gain per experiment by
    spanning orthogonal physicochemical axes: net charge (positive vs
    negative), hydrophobicity (aliphatic vs aromatic), polar H-bonding (Ser/
    Thr vs Asn/Gln), conformational rigidity/flexibility (Pro/Gly), redox/
    metal coordination (Cys/His, D-x-D), and secondary-structure propensities
     (Ala/Leu/Ile/Val).
- The set below implements a fractional-factorial-like sweep across these
    axes, so early results can reveal main effects and simple interactions by
     comparing closely related contrasts (e.g., cationic vs anionic; aromatic
     vs aliphatic; HXH vs D-x-D; rigid PxxP vs flexible PGPG).
- Prioritization favors maximal diversity first, then close contrasts to
    refine mechanistic hypotheses.

Priority ranking with brief chemical justification
1) KRRR âĂŤ Strongly cationic, Arg/Lys-rich for electrostatic binding to
    acidic/negatively charged targets (e.g., nucleic acids or acidic protein
    patches).
2) DEED âĂŤ Strongly anionic, acidic cluster probe; contrasts directly with
    KRRR to test electrostatic preference.
3) WFWY âĂŤ Aromatic-rich, tests piâĂŞpi/piâĂŞcation interactions and
    hydrophobic packing using mixed F/W/Y.
4) LVIL âĂŤ Aliphatic hydrophobic core motif; contrasts WFWY to separate
    aromatic-specific effects from generic hydrophobics.
5) STST âĂŤ Ser/Thr H-bond network and turn/helix edge propensity; probes
    polar OH-mediated recognition.
6) HEGH âĂŤ HExH-like histidine motif; tests metal coordination and pH-
    sensitive binding via His.
7) DADE âĂŤ D-x-D acidic motif; canonical divalent metal-binding/chelation
    contrast to HEGH.
8) CAGC âĂŤ Cys pair (CxxC) enables redox/disulfide or soft-metal
    coordination; probes thiol chemistry and structural constraint.
9) PGPG âĂŤ Alternating Pro/Gly for high turn propensity and conformational
    flexibility; tests if tight turns are favored.
10) PAPP âĂŤ PxxP motif to enforce rigidity and SH3-like motif geometry;
    contrasts PGPG for flexibility vs rigidity.
11) RKRK âĂŤ Alternating cationic pattern to test charge spacing vs clustered
     cationic (KRRR).
```

```
12) QNQQ âĂŤ Amide-rich polar H-bonding without charge; probes polar
    recognition distinct from Ser/Thr chemistry.
13) FWLW âĂŤ Alternating aromatic/aliphatic to test interleaved hydrophobic/
    aromatic patterns vs clustered aromatics (WFWY).
14) AAAA âĂŤ Helix-prone neutral baseline; a clean reference for main-effect
    comparisons across positions.
15) LYKS âĂŤ Mixed motif (hydrophobic-aromatic-basic-polar) to sample
    heterogeneous interfaces and potential piâĂŞcation (Y/K) interplay.

Final list of 15 sequences for next-cycle validation (top 10 prioritized
    above):
KRRR, DEED, WFWY, LVIL, STST, HEGH, DADE, CAGC, PGPG, PAPP, RKRK, QNQQ, FWLW,
     AAAA, LYKS
```

```
The validation experiment in cycle 0 is finished. These are the results:
sequence  fitness
    AAAA    0.074
    KRRR    0.045
    LVIL    0.020
    DADE    0.016
    WFWY    0.013
    STST    0.013
    PAPP    0.010
    RKRK    0.002
    DEED    0.000
    HEGH    0.000
```

```
Cycle 1 analysis and hypotheses from cycle 0 data
- Small/neutral wins: AAAA is top (0.074), suggesting minimal side-chain bulk
     and strong helix propensity (Ala) are beneficial. This points toward a
    preference for compact, low-steric side chains and possibly an alpha-
    helical segment.
- Cationic helps, especially Arg-rich clusters: KRRR is second-best (0.045),
    while alternating cationic RKRK is poor (0.002). This implies clustered
    or context-dependent cationic interactions are beneficial; Arg appears
    superior to Lys (guanidinium vs ammonium).
- Hydrophobic tolerated but suboptimal: LVIL (0.020) and WFWY (0.013)
    indicate hydrophobic packing/aromatics help but are inferior to A-rich.
- Acidic clusters and His motifs are disfavored: DEED (0.000), HEGH (0.000);
    DADE (0.016) shows that isolated/alternating acidity with small residues
    can be tolerated, but excess negative charge is detrimental.
- Polar OH acceptable but not optimal: STST (0.013) suggests limited benefit
    from Ser/Thr networks.

Working hypotheses for next designs
1) Ala-rich background is optimal; adding limited positive charge (preferably
     Arg) may boost fitness if positioned correctly. Test single-position Arg
     and Lys scans on AAAA.
2) Arg likely outperforms Lys; quantify this across positions.
3) Two Arg in an Ala background may be better than one but worse than fully
    charged; test spacing effects (AARR).
4) Small/polar (Ser) or smallest (Gly) substitutions might be position-
    sensitive; edge vs internal effects may differ.
```

```
5) Keep exploration limited for acidic/aromatic features; include one
    âĂIJsentinelâĂİ acidic variant to measure penalty of an isolated negative
    charge in an Ala carpet.

Priority ranking with justification
1) RAAA âĂŤ Single Arg at position 1 on AAAA; probes N-terminal placement of
    cationic charge with minimal steric penalty.
2) ARAA âĂŤ Single Arg at position 2; tests internal Arg placement effect.
[...]
```

```
The validation experiment in cycle 1 is finished. These are the results:
sequence  fitness
    AAAG    0.279
    AAAS    0.221
[...]
```

```
Cycle 2 analysis and mechanistic hypotheses
- Strong terminal preferences emerged:
  - Position 4: Gly and Ser are highly beneficial (AAAG 0.279 > AAAS 0.221 >>
     AAAA 0.074). This is signature of a C-terminal cap preference for small/
    flexible or H-bonding residues that relieve steric clash and/or stabilize
     a local turn/helix cap (G > S; next likely T, then Asn; bulkier polar
    like Gln expected weaker).
  - Position 1: Gly improves fitness (GAAA 0.119 > AAAA 0.074). N-terminal
    flexibility or reduced steric bulk appears favorable.
- Charges are generally detrimental, especially Arg at any position (RAAA,
    ARAA, AAAR all ~0âĂŞ0.01). Lys is tolerated only internally (AAKA âĹĹ
    AAAA), but not beneficial; Lys at pos1 is harmful (KAAA 0.018). Acidic
    residues are strongly disfavored (AADA 0.000). Hydrophobics/aromatics
    also underperform.
- Therefore, the productive design space is: small/non-bulky residues,
    especially G/S at termini; possibly T/N as alternative C-caps; internal
    positions likely prefer A/S/G with caution about excessive flexibility
    from multiple Gly.

Design strategy for Cycle 2
- Test additivity/epistasis by combining the best terminal features: G/S at
    pos4 with G (and S) at pos1.
- Expand C-terminal cap chemistry: Thr and Asn; include Gln as a bulkier
    polar comparator.
- Probe whether added flexibility internally helps or hurts: single Gly scans
     at positions 2 and 3; single Ser scans at positions 2 and 3.
- Limit risky motifs (charge, aromatics, strong hydrophobics), reserving
    budget for the productive small/polar space.

Priority ranking with chemical justification
1) GAAG âĂŤ Combines the two strongest single effects (G1 + G4); tests near-
    maximal flexibility at termini; highest expected fitness.
2) GAAS âĂŤ G1 + S4; probes if H-bond cap at 4 with flexible N-terminus
    rivals G1/G4.
[...]
```

```
<O>The validation experiment in cycle 2 is finished. These are the results:
sequence  fitness
    GAAS    0.315
    SAAG    0.238
[...]
```

```
Cycle 3 analysis and mechanistic hypotheses
- Clear SAR emerged: termini prefer very small/polar residues, centers prefer
     Ala.
  - C-terminal cap: S4 and G4 are strongly beneficial (AAAS 0.221; AAAG
    0.279). Thr4 is highly deleterious (AAAT 0.008). Asn4 is roughly neutral/
    slightly worse than baseline (AAAN 0.075).
  - N-terminal: G1 improves baseline (GAAA 0.119). S1 is promising when
    combined with favorable C-caps (SAAG 0.238; SAAS 0.186), but S1 alone is
    unmeasured.
  - Best family identified: X A A Y with X âĹĹ {G,S}, Y âĹĹ {G,S}; strongest
    observed GAAS (0.315) > SAAG (0.238) > GAAG (0.202) > SAAS (0.186),
    indicating terminal epistasis (S4 best with G1; G4 best with S1).
  - Internal positions penalize flexibility/polarity: introducing Gly/Ser at
    2 or 3 reduces fitness (AGAA 0.071; AAGA 0.023; AASA 0.036), though a
    strong C-cap can partially rescue (AAGS 0.107). Centers prefer Ala.

Working hypotheses to guide next designs
1) The global maximum likely resides near XAA Y with X,Y âĹĹ {G,S}; we will
    probe whether modest, chemically plausible deviations at positions 2âĂŞ3
    (single Ser/Gly) can fine-tune the optimum under favorable terminal pairs
    .
2) N-capping alternatives: Pro or Asn at position 1 might sometimes stabilize
     local turns/caps; test them in the context of strong C-caps (S4/G4).
3) Cys as a small polarizable C-cap surrogate for Ser/Gly: test C4 with/
    without G1 to map tolerance. Include one Gln4 sentinel to bound amide-cap
     performance.
4) Complete missing single-mutant baselines: SAAA (S1 alone) and ASAA (S at
    pos2) for positional sensitivity.

Priority ranking with justification
[...]
```

```
The validation experiment in cycle 3 is finished. These are the results:
sequence  fitness
    GPAS    0.341
[...]
```

```
Cycle 4 strategy and chemical reasoning
- Confirmed terminal epistasis: Position 4 strongly prefers S or G but with
    context. G1 pairs best with S4 (GAAS 0.315 > GAAG 0.202); S1 pairs best
    with G4 (SGAG 0.277 > SAAS 0.186). Amide caps (N/Q at 4) and Thr4 are
    poor; Cys4 is mediocre. Keep 4 âĹĹ {S,G}.
- Internal positions favor Ala; introducing Gly or Ser at 2 or 3 depresses
    fitness even under favorable termini (e.g., GGAS 0.100, GAGS 0.093, GSAS
    0.085). Key exception: Proline at position 2 gives a real boost on the G1
```

```
    /S4 ridge (GPAS 0.341 > GAAS 0.315), likely via local turn/helix capping
    and preorganization without over-flexibilizing the core.
- N-terminus: G1 and S1 are both good in the right context; S1-G4 and G1-S4
    look like complementary optima. Nonpolar/hydrophobic or charged
    substitutions were uniformly harmful earlier.

Hypotheses to test now
1) Proline-at-2 as a transferable enhancer: Move P2 to the S1/G4 ridge to
    create SPAG; also test if P2 improves the weaker G1/G4 ridge (GPAG) and
    the weaker S1/S4 ridge (SPAS). Expect SPAG to be competitive with the
    current best.
2) Is P best kept at 2? Shift P to position 3 under strong terminals to
    quantify penalty vs benefit (GAPS, GAPG, SAPG), anticipating lower but
    mapping robustness.
3) N-cap alternatives: Thr1 can N-cap in helices; test TPAG/TPAS as
    conservative variants against S1 with both C-termini.
4) How much turn bias is too much? Introduce P2-G3 tight turn (GPGS, SPGS)
    and double-Pro core (GPPS, GPPG) as stress tests; expect reduced but
    informative.
5) Is G1 essential for P2 benefit? Compare APAS vs GPAS baseline (already
    known) to quantify the necessity of the G1 N-cap in the P2 motif.
6) Internal polarity under P2: Test GPSS to confirm S3 remains penalizing
    even with P2.

Priority ranking with justification
1) SPAG âĂŤ Aligns with S1/G4 synergy; adds P2 which helped on G1/S4. Highest
     likelihood to meet or exceed current best.
[...]
```

```
The validation experiment in cycle 4 is finished. These are the results:
sequence  fitness
    TPAG    0.292
[...]
```

```
Cycle 5 chemical reasoning and goals
- Clear terminal epistasis remains: position 4 prefers S or G, but the
    preference flips with N-terminus and with the presence of Pro at position
     2.
  - With G1 and P2 present, S4 is superior (GPAS 0.341 > GPAG 0.225).
  - With T1 and P2, G4 is far superior (TPAG 0.292 >> TPAS 0.113).
  - With S1 and P2, evidence from G3 sentinels suggests G4 > S4 (SPGG 0.172 >
     SPGS 0.071); we now test this in the optimal A3 background.
- P2 is a key enhancer when used with optimal terminals; P3 is deleterious.
    Internal polar/flexible residues at 2âĂŞ3 are generally harmful unless
    they are precisely Pro2 or Gly2 in the S1/G4 context (SGAG 0.277 > SAAG
    0.238).
- A3 appears optimal; G3/S3 consistently reduce fitness, but we will probe
    whether a small hydrophobe (V3) can be tolerated/improve preorganization
    under strong termini/P2.
- N-terminal identity strongly modulates the best C-cap when P2 is present;
    we will systematically map X1 in {S,A,N} across Y4 in {G,S} with P2 fixed
     to locate the global maximum near the GPAS/SPAG/TPAG ridge.

Priority ranking with justification
```

```
1) SPAG âĂŤ Tests S1/P2 with G4; predicted to be high due to S1âĂŞG4 synergy
    and P2 enhancement (cf. SGAG 0.277; P2 likely boosts further).
[...]
```

```
The validation experiment in cycle 5 is finished. These are the results:
sequence  fitness
    GPVS    0.495
[...]
```