AUTOSCIENTISTS: Self-Organizing Agent Teams for Long-Running Scientific Experimentation

Shanghua Gao*
Harvard University
shanghua_gao@hms.harvard.edu

Ada Fang*
Harvard University
ada_fang@g.harvard.edu

Marinka Zitnik
Harvard University
marinka@hms.harvard.edu

AUTOSCIENTISTS website: https://autoscientists.openscientist.ai
AUTOSCIENTISTS code: https://github.com/mims-harvard/AutoScientists

Abstract

Scientific research proceeds through iterative cycles of hypothesis generation, experiment design, execution, and revision. AI agents can automate parts of this process, but existing approaches typically follow a single research trajectory or coordinate through a central planner with fixed objectives. As a result, they struggle to sustain parallel exploration, adapt as experimental evidence changes, or preserve knowledge of failed directions over long-running experiments. We introduce AUTOSCIENTISTS, a decentralized team of AI agents for long-running computational scientific experimentation. Agents interpret a shared experimental state, self-organize into teams around promising hypotheses, critique proposals before using experimental compute, and share successes and failures to reduce redundant exploration. Under matched experimental budgets, AUTOSCIENTISTS improves over prior AI agents across biomedical machine learning, language-model training optimization, and protein fitness prediction. On BioML-Bench, spanning biomedical imaging, protein engineering, single-cell omics, and drug discovery, AUTOSCIENTISTS achieves a mean leaderboard percentile of 74.4% across 24 tasks, improving over the strongest AI agent by +8.33%. On GPT training optimization, AUTOSCIENTISTS reaches a target validation bits-per-byte 1.9× faster than Autoresearch and continues discovering improvements from a starting champion where the single-agent approach finds none (7 vs. 0 accepted improvements). On ProteinGym fitness prediction, AUTOSCIENTISTS discovers a method for ACE2-Spike binding that improves over the current state-of-the-art model by +12.5% in Spearman correlation. Applied without modification across all 217 ProteinGym assays, the same method improves over the prior state of the art by +6.5% (Spearman correlation).

1 Introduction

AI agents for science are beginning to move beyond answering questions and running predefined workflows toward proposing and executing research steps [1], from protein engineering in biology to language model optimization in machine learning [2, 3]. Agents can generate hypotheses, synthesize literature, design computational experiments, write and execute code, and refine models from experimental feedback [4–10]. However, most current approaches remain limited to short-horizon

*Equal contribution.

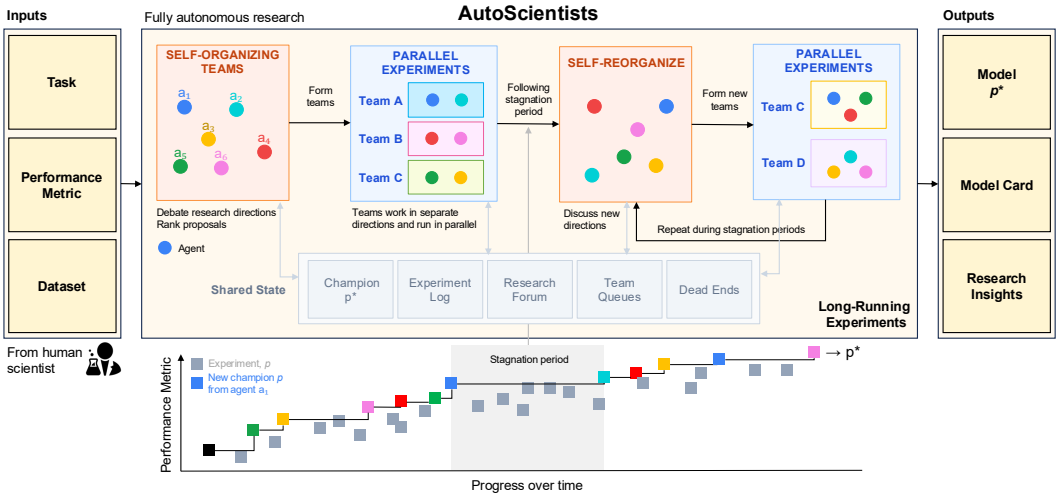


Figure 1: **Self-organizing agent teams for long-running experimentation.** Overview of AUTOSCIENTISTS. Agents identify promising research directions, organize into teams, and execute experiments in parallel.

optimization or fixed pipelines. They typically follow a single reasoning thread or use a search-space decomposition set at the start of the run. This assumption breaks down in long-running scientific experimentation, where research directions are not known in advance and change over time.

Existing AI agents can run experiments, but long-running science requires more: maintaining competing hypotheses, updating them as evidence changes, and using failures to redirect the search. Single-agent systems such as AIDE [11] and Autoresearch [3] iteratively refine proposals but follow a single search trajectory, limiting their ability to explore competing hypotheses in parallel. Multi-agent systems [12–14] distribute work across agents, but still coordinate through a central structure: a planner decomposes the problem, a search algorithm ranks proposals, or agents converge through discussion or voting [15, 16]. These approaches assume that the search space can be partitioned into stable directions at the start of the run. *In long-running experimentation, however, productive directions shift as evidence accumulates. Some hypotheses stop yielding improvements, failed directions must be tracked to avoid repeated exploration, and new hypotheses often emerge only after earlier experiments are analyzed.*

Present Work. We introduce AUTOSCIENTISTS, a self-organizing agent team for long-running scientific experimentation that coordinates without a central orchestrator agent (Figure 1). Rather than receiving assignments from a planner, agents act on a shared state that records proposals, experiments, results, failures, and the current champion. Teams form dynamically through agent interaction rather than user-specified decomposition. Agents post experiment proposals to a shared forum, where peers critique them before execution, filtering weak ideas before compute is committed. As results accumulate, agents reorganize around productive directions, retire exhausted directions, and share successes and failures across teams to reduce redundant exploration. We apply AUTOSCIENTISTS to research tasks spanning imaging, drug discovery, single-cell omics, protein engineering, protein fitness prediction, and language model training optimization.

Across benchmarks, AUTOSCIENTISTS improves over existing AI agents. On BioML-Bench [2], AUTOSCIENTISTS achieves the highest average leaderboard percentile among the evaluated agents, reaching 74.40% across 24 biomedical ML tasks compared with 66.07% for Autoresearch under the same task interface, model backend, and hardware budget. The performance improvements are largest in drug discovery, where AUTOSCIENTISTS improves from 46.16% to 64.52%. On GPT nanochat training optimization [3], AUTOSCIENTISTS reaches the same intermediate validation loss in 34 experiments that Autoresearch reaches in 65 experiments, and when continuing from a AUTOSCIENTISTS champion reaches a validation bits-per-byte (bpb) of 0.9730 while Autoresearch finds no accepted improvements over 100 experiments. On ProteinGym supervised substitution fitness prediction [17], AUTOSCIENTISTS starts from Kermut and discovers a Kermut extension that improves ACE2–Spike binding Spearman's ρ from 0.747 to 0.840. Furthermore, the frozen recipe transfers across the full 217-assay ProteinGym supervised substitution benchmark, improving the official average Spearman's ρ from 0.657 to 0.700. Below we summarize our contributions:

- **A self-organizing agent team for long-horizon scientific experimentation**. Unlike prior systems that rely on central coordinators, consensus-based discussion, or fixed decompositions of the search space, AUTOSCIENTISTS allows agents to independently interpret a shared experimental state and decide which hypotheses to pursue. Agents post proposals to a shared forum where peers critique and filter them before experiments run, allowing teams and experimental directions to emerge through interaction rather than external assignment.
- **State-of-the-art performance across scientific domains, with sustained improvement during long-running experimental search.** AUTOSCIENTISTS improves over prior agents on biomedical ML, protein fitness prediction, and language-model training optimization, and continues identifying productive modifications after single-agent baselines stop improving.

2 Related Work

AI Agents for Scientific Research. AI agents are increasingly being developed to automate scientific workflows, including literature review, hypothesis generation, tool use, code execution, experimental design, benchmarking, and manuscript drafting [18–22]. Biomedical agents combine multi-step reasoning with biomedical tools, literature grounding, omics analysis, code execution, and evidence reconciliation [10, 6, 9, 23–25]. Other systems push toward longer-horizon discovery through repeated cycles of literature search, hypothesis generation, debate, refinement, tool integration, optimization, equation discovery, self-directed exploration, and skill accumulation [4, 7, 26–31, 12, 32]. Several scientific-agent systems rely on role-specialized architectures, such as PI–scientist–critic organizations or Manager–Developer–Critic–Tool Creation pipelines [14, 5]. In AI research, related systems have also generated research papers or evolved algorithms through iterative code modification and experimentation [33, 3, 34]. Our system differs in that agents collectively determine research directions through discussion and coordinate through shared forums rather than fixed pipelines or a central orchestrator that directs others. Unlike debate frameworks that use discussion to converge on a shared hypothesis [15, 16], AUTOSCIENTISTS uses discussion to filter out weak proposals before any experiment runs, while allowing agents to continue pursuing different research directions in parallel.

Coordination of Multi-Agent Systems. Beyond scientific applications, multi-agent performance depends strongly on collaboration structure and agent composition [35–40]. Interaction is not automatically beneficial. For example, multi-agent systems have underperformed their best individual member on tasks [41, 40], and recent benchmarks analyse how collaboration and competition affect collective performance [42]. These findings motivate our ablation studies and comparison to single-agent baselines like Autoresearch. Human scientific teams provide a complementary perspective as they benefit from diversity and flatter structures, but excessive diversity can introduce coordination costs [43–46]. Recent work also emphasizes context management, memory, and reusable skills for sustained collaboration [12, 47, 32]. Our system draws on these findings by organizing agents as teams focused on complementary research directions and uses shared forums to support conference-style knowledge sharing and collective intelligence [48].

3 AUTOSCIENTISTS: Long-Running Self-Organizing Agent Teams

We proceed by formalizing long-running scientific experimentation as an iterative search process and introduce AUTOSCIENTISTS. We first define the optimization setting and then describe how agents organize into teams, propose and execute experiments, exchange experimental evidence through a shared state, and reorganize as search trajectories evolve over time.

3.1 Problem Formulation

We are given a task description, optionally accompanied by an initial program (e.g., a training script) p_0, together with a dataset $\mathcal{D}$ and an evaluation metric ℓ.

The dataset $\mathcal{D}$ consists of a training set $\mathcal{D}_{\text{train}}$ and an evaluation protocol. The evaluation protocol may take one of the following forms: a validation set $\mathcal{D}_{\text{val}}$, or a cross-validation (CV) scheme over $\mathcal{D}_{\text{train}}$. We denote by $\ell_{\text{eval}}(p; \mathcal{D})$ the evaluation metric computed under this protocol.

A system of n long-running LLM agents $\mathcal{A} = \{a_1, \ldots, a_n\}$ iteratively proposes and generates new programs. Long-running agents a_i persist over the course of the search process, maintaining internal

state and updating their behavior based on accumulated experience. This contrasts with one-shot agents that generate a solution in a single forward pass. Each proposed program p is trained on $\mathcal{D}_{\text{train}}$ and evaluated using ℓ_{eval}. The goal is to identify a program

$$p^* = \arg\max_{p \in \mathcal{P}} \ell_{\text{eval}}(p; \mathcal{D}),$$

where $\mathcal{P}$ denotes the space of programs explored by the agents during the search process, optionally initialized from p_0. We assume without loss of generality that ℓ is oriented so that higher values correspond to better performance (e.g., by negating metrics that are typically minimized such as loss).

At the end of the search process, performance is reported using ℓ_{test} if a held-out test set is available, otherwise, ℓ_{eval} is used (e.g., validation or CV performance).

3.2 AUTOSCIENTISTS Approach

Overview. AUTOSCIENTISTS deploys n long-running agents that maintain state across the run, adapt their search strategy, self-reorganize into teams, and update their search behavior from accumulated evidence (Figure 1). The system alternates between two phases. In the *discussion* phase, agents analyze the task, propose experimental directions, and organize into teams. In the *execution* phase, teams run parallel experiments and write results back to the shared state $\mathcal{S}$. When performance on ℓ_{eval} stagnates, agents reopen discussion and may reorganize teams around different directions. This cycle continues for the duration of the run and is coordinated through $\mathcal{S}$ rather than a central planner agent. Each agent uses an LLM, so AUTOSCIENTISTS approach is LLM-agnostic.

Discussion and Self-Organization. Agents identify and revise research directions through discussion phases, without a predefined partition of the search space. AUTOSCIENTISTS initializes with no teams and no predefined directions. At the start of each discussion phase, all agents read the task specification, the current champion p^*, and prior posts on the shared forum $\mathcal{F}$. Discussion proceeds over multiple rounds. Early rounds focus on proposing and evaluating candidate directions: agents independently analyze p^*, propose modifications, critique competing proposals, and identify gaps in the search space. Later rounds organize agents into K teams $\{\mathcal{T}_1, \ldots, \mathcal{T}_K\}$, where each team is assigned one research direction. The final agent in the discussion round consolidates the proposals into a roster $R = \{(\mathcal{T}_k, \text{axis}_k, \text{members}_k)\}_{k=1}^{K}$ and writes it to $\mathcal{S}$. Subsequent agents adopt the roster on their next heartbeat.

The roster changes as evidence accumulates. When a team stops producing improvements, agents trigger a new discussion phase and review results across all teams. Through the shared research forum, agents can propose to create, merge, split, or rebalance teams, with changes requiring endorsement from affected teams before taking effect. This allows AUTOSCIENTISTS to redirect effort during the run: exhausted directions can be retired, and newly emerging hypotheses can form new teams.

Long-Running Parallel Experiments. Each team $\mathcal{T}_k$ operates a continuous propose-execute loop. Every agent runs a heartbeat cycle: read the shared state $\mathcal{S}$, act according to its role, write results back to $\mathcal{S}$, repeat. Agents persist across cycles with their own identity and memory files, accumulating knowledge over the duration of the run. Two specialized roles collaborate in each team:

(1) Analyst Agents. Analysts maintain the team's search knowledge and propose experiments. Each heartbeat cycle, an analyst reads the experiment log $\mathcal{L}$, audits which research directions have never been tested, and posts proposals to the team queue Q_k. Proposals are ranked by observed effect sizes from $\mathcal{L}$, where underexplored research directions are prioritized, and research directions with consistently small effects are deprioritized (details in Appendix A.7). After a champion update, the analyst identifies what features made the improvement and proposes variants that share the same features.

(2) Experiment Agents. Experiment agents claim experiments from the team queue Q_k, apply the code change to p^*, train, and record the outcome to $\mathcal{L}$ and $\mathcal{F}$. Since the evaluation metric ℓ may be stochastic (e.g., variation due to random seed of training runs), improvements within the empirically measured noise band are confirmed on a second seed before promotion to p^* (details in Appendix A.6). All results, including failures, are visible to every agent across all teams.

Teams execute in parallel for the full duration. As experiments accumulate, teams track failed experiments in a dead-end registry $\mathcal{D}_k$ to avoid repeating unproductive directions, and rank its queue Q_k by observed effect sizes from $\mathcal{L}$ so that underexplored directions are tried first. When a team's

recent experiments consistently fail to improve p^* (e.g., no improvement in the last 10 experiments), agents return to discussion and may reorganize into new teams around more productive directions.

Shared State. The system maintains a shared state accessible to all agents, consisting of four layers: a champion p^* tracking the current best model with full hyperparameters and reproduction instructions; an experiment log $\mathcal{L}$ of every completed experiment with outcome, metric delta, and training diagnostics; a shared forum $\mathcal{F}$ of structured posts where proposals are debated, results announced, and mechanistic analyses shared; and team-local state (per-team experiment queues Q_k, dead-end registries $\mathcal{D}_k$, and hypothesis documents) that is readable cross-team. Details are in Appendix A.

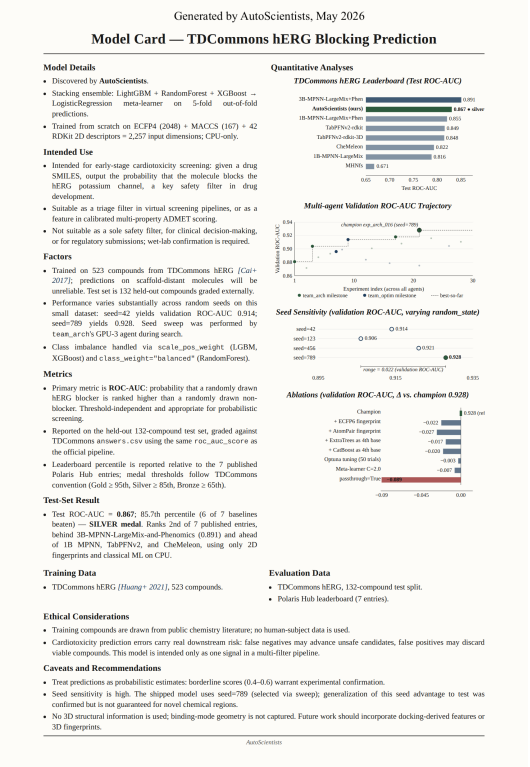
Output. AUTOSCIENTISTS outputs the final champion model p^* together with a model card and a research findings report derived from the agents' experimental process. AUTOSCIENTISTS produces the main technical components of a model card [49]. Model architecture, hyperparameters, and training procedures are recorded in the reproducible champion training script. Training and evaluation datasets are inherited from the shared task specification, and quantitative performance metrics are stored in the champion record. Figure 2 shows a model card for a hERG prediction model discovered by AUTOSCIENTISTS on BioML-Bench.

In addition to the final model, AUTOSCIENTISTS records the experimental search process that produced it. Dead-end registries store failed experimental directions together with the tested axis, research direction, performance change, and rejection reason. Analyst agents document the mechanisms underlying successful modifications and propose related follow-up directions. Combined with the full experiment log, these artifacts provide a record of how hypotheses evolved during the run, which directions were abandoned, and how the final model emerged from accumulated experimental evidence. Appendix E presents the complete set of artifacts produced by AUTOSCIENTISTS on the GPT nanochat task.

4 Experiments

4.1 Implementation Details

All agents in AUTOSCIENTISTS use the same base model, Claude Code coding agent [50] with the base LLM Claude Sonnet 4.6 [51]. We use the same model backend for AUTOSCIENTISTS and the Autoresearch baseline. Each agent is repeatedly invoked by a deterministic monitor process in a heartbeat loop. AUTOSCIENTISTS was given access to H100 GPUs for running experiments. For further details on reproducing experimental results refer to Appendix 6. Unless specified otherwise, the AUTOSCIENTISTS team is composed of 3 analyst agents and 6 experiment agents.


Generated by AutoScientists, May 2026

Model Card — TDCommons hERG Blocking Prediction

Model Details
- Discovered by **AutoScientists**.
- Stacking ensemble: LightGBM + RandomForest + XGBoost → LogisticRegression meta-learner on 5-fold out-of-fold predictions.
- Trained from scratch on ECFP4 (2048) + MACCS (167) + 42 RDKit 2D descriptors = 2,257 input dimensions; CPU-only.

Intended Use
- Intended for early-stage cardiotoxicity screening: given a drug SMILES, output the probability that the molecule blocks the hERG potassium channel, a key safety filter in drug development.
- Suitable as a triage filter in virtual screening pipelines, or as a feature in calibrated multi-property ADMET scoring.
- Not suitable as a sole safety filter, for clinical decision-making, or for regulatory submissions; wet-lab confirmation is required.

Factors
- Trained on 523 compounds from TDCommons hERG *[Cai+ 2017]*; predictions on scaffold-distant molecules will be unreliable. Test set is 132 held-out compounds graded externally.
- Performance varies substantially across random seeds on this small dataset: seed=42 yields validation ROC-AUC 0.914; seed=789 yields 0.928. Seed sweep was performed by `team_arch`'s GPU-3 agent during search.
- Class imbalance handled via `scale_pos_weight` (LGBM, XGBoost) and `class_weight="balanced"` (RandomForest).

Metrics
- Primary metric is **ROC-AUC**: probability that a randomly drawn hERG blocker is ranked higher than a randomly drawn non-blocker. Threshold-independent and appropriate for probabilistic screening.
- Reported on the held-out 132-compound test set, graded against TDCommons `answers.csv` using the same `roc_auc_score` as the official pipeline.
- Leaderboard percentile is reported relative to the 7 published Polaris Hub entries; medal thresholds follow TDCommons convention (Gold ≥ 95th, Silver ≥ 85th, Bronze ≥ 65th).

Test-Set Result
- Test ROC-AUC = **0.867**; 85.7th percentile (6 of 7 baselines beaten) — **SILVER medal**. Ranks 2nd of 7 published entries, behind 3B-MPNN-LargeMix-and-Phenomics (0.891) and ahead of 1B MPNN, TabPFNv2, and CheMeleon, using only 2D fingerprints and classical ML on CPU.

Training Data
- TDCommons hERG *[Huang+ 2021]*, 523 compounds.

Evaluation Data
- TDCommons hERG, 132-compound test split.
- Polaris Hub leaderboard (7 entries).

Ethical Considerations
- Training compounds are drawn from public chemistry literature; no human-subject data is used.
- Cardiotoxicity prediction errors carry real downstream risk: false negatives may advance unsafe candidates, false positives may discard viable compounds. This model is intended only as one signal in a multi-filter pipeline.

Caveats and Recommendations
- Treat predictions as probabilistic estimates: borderline scores (0.4–0.6) warrant experimental confirmation.
- Seed sensitivity is high. The shipped model uses seed=789 (selected via sweep); generalization of this seed advantage to test was confirmed but is not guaranteed for novel chemical regions.
- No 3D structural information is used; binding-mode geometry is not captured. Future work should incorporate docking-derived features or 3D fingerprints.

AutoScientists

Figure 2: **Model card produced by AUTOSCIENTISTS.** TDC hERG Blocking Prediction model discovered by AUTOSCIENTISTS.

4.2 End-to-End Biomedical Machine Learning with AUTOSCIENTISTS

Setup. We evaluate AUTOSCIENTISTS on BioML-Bench, a benchmark of 24 end-to-end biomedical machine-learning tasks spanning biomedical imaging (4), drug discovery (9), protein engineering (6), and single-cell omics (5) [2]. Each task provides a natural-language task description, training data, test inputs, and an example submission format. For each of the four task types, an LLM-generated general model paradigm menu is included in the AUTOSCIENTISTS agent's discussion prompts to encourage

Table 1: BioML-Bench comparison of Biomni, Autoresearch, and AUTOSCIENTISTS under matched per-domain experimental compute budgets. Values are mean (SE) and rank among the three agents.

Domain	Biomni		Autoresearch		AUTOSCIENTISTS	
	Leaderboard ↑	Rank ↓	Leaderboard ↑	Rank ↓	Leaderboard ↑	Rank ↓
Biomedical Imaging ($n = 4$)	19.04 (10.83)	3.00	39.60 (21.75)	1.75	**45.75 (22.18)**	**1.25**
Drug Discovery ($n = 9$)	47.91 (10.77)	2.22	46.16 (10.59)	2.00	**64.52 (8.37)**	**1.78**
Protein Engineering ($n = 6$)	93.94 (3.83)	2.50	**96.97 (3.03)**	2.00	**96.97 (3.03)**	**1.50**
Single Cell Omics ($n = 5$)	78.00 (10.20)	2.60	86.00 (9.80)	1.80	**88.00 (9.70)**	**1.60**

Full metrics are reported in Table S6.

diverse research directions. AUTOSCIENTISTS develops models using the task description, training data, and development-time validation feedback. Hidden test labels and private grader files are kept outside the agent workspace and are accessed only by the external evaluator. Following BioML-Bench, we report four task-level outcomes: leaderboard percentile relative to public human submissions, whether the submission exceeds the public leaderboard median, whether it receives any medal, and completion rate. We compare against the published BioML-Bench results for Reference, MLAgentBench [52], AIDE [11], STELLA [5], and Biomni [10]. We additionally adapt Autoresearch [3], implemented with the same coding-agent backend, to the BioML-Bench task. Experimental compute settings and task-specific setup details are provided in Appendix F.

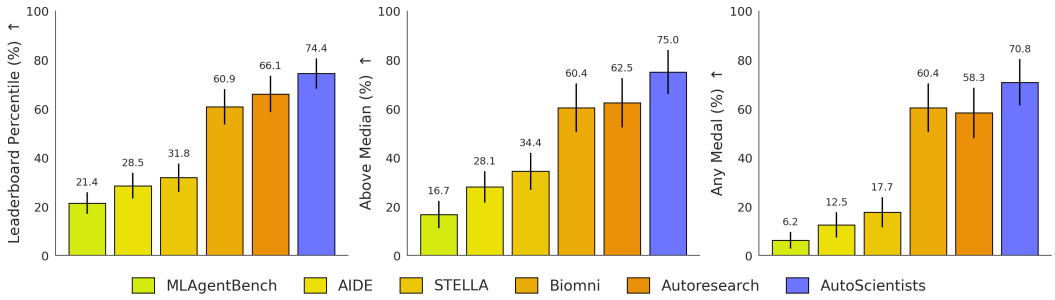


Figure 3: **AUTOSCIENTISTS improves performance across BioML-Bench tasks.** Performance on 24 biomedical tasks measured by leaderboard percentile (**left**), proportion above the public leaderboard median (**middle**), and proportion awarded a medal (**right**). Error bars show standard error of the mean. Additional results are reported in Table S6.

Results. We report aggregate performance in Fig. 3 and domain-level performance in Table 1. Overall, AUTOSCIENTISTS achieves the highest mean (SE) leaderboard percentile among the evaluated systems, with 74.40 (6.20)% compared with 66.07 (7.38)% for Autoresearch, a gain of +8.33 leaderboard-percentile points. AUTOSCIENTISTS completes all 24 tasks. AUTOSCIENTISTS shows the strongest gain in drug discovery, reaching 64.52 (8.37)% mean (SE) leaderboard percentile compared with 47.91 (10.77)% for Biomni. Protein engineering is the strongest domain in absolute leaderboard percentile, but it is also largely saturated with both AUTOSCIENTISTS and Autoresearch obtaining 96.97 (3.03)%, although AUTOSCIENTISTS achieves a better mean rank of 1.50. Instead, Sec. 4.4 evaluates the more relevant question of whether AUTOSCIENTISTS can discover a single method that transfers across the full ProteinGym supervised substitution benchmark. Biomedical imaging remains the most challenging domain and each task requires substantially larger image-model training. We summarize the final p^* AUTOSCIENTISTS-approaches in Appendix F.5. To complement the quantitative results, we inspected the shared state and agent logs of AUTOSCIENTISTS to determine whether deliberation changed the experiments selected for execution. Fig. 5 shows representative examples in which agents diversified away from redundant proposals, identified saturated search directions, transferred hypotheses across teams, and retired dead-end directions after stagnation, supporting the interpretation that AUTOSCIENTISTS improves experiment selection under a fixed experiment compute budget.

4.3 GPT Training Optimization with AUTOSCIENTISTS

We next evaluate whether AUTOSCIENTISTS generalizes beyond biomedical tasks by applying it to GPT nanochat training optimization, the language-model training benchmark introduced by Autoresearch [3]. In this task, each experiment modifies a training program and is evaluated by the resulting validation bits-per-byte, so progress depends on selecting useful changes under a fixed experimental budget. This setting tests whether AUTOSCIENTISTS can coordinate search over interacting choices in architecture, optimization, and training schedule.

Setup. Each experiment is a single 5-minute GPT training run on one H100 GPU, evaluated by validation bits-per-byte (`val_bpb`; lower is better). We compare AUTOSCIENTISTS against single-agent Autoresearch [3] on the same code repository. Here the only variable is orchestration. We evaluate two regimes: (i) *From Autoresearch baseline*: both systems start from the same nanochat code at `val_bpb` $= 0.998$ and search for improvements, (ii) *From a* AUTOSCIENTISTS *champion*: both systems start from the champion p^* obtained after 50 prior AUTOSCIENTISTS experiments (`val_bpb` $= 0.9777$) and are given identical access to the negative-knowledge file `EXPLORED.md` listing previously dead-ended directions. We report per-experiment trajectories rather than wall clock so that the comparison isolates orchestration from hardware allocation. Both regimes are compared in Appendix B.1.

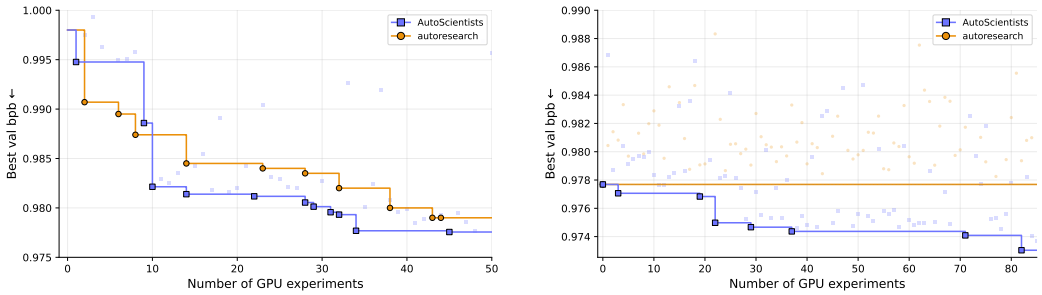
From Autoresearch Baseline. In the from-baseline regime (Fig. 4a), AUTOSCIENTISTS reaches `val_bpb` ≈ 0.978 in 34 experiments, while single-agent Autoresearch reaches the same value only after 65 experiments. At this target loss, AUTOSCIENTISTS uses $1.9\times$ fewer experiments. Both methods improve quickly in the first ten experiments and their best-so-far `val_bpb` curves stay within $\sim$0.005 of each other; from experiment 10 onward, AUTOSCIENTISTS pulls ahead and stays strictly lower for the remainder of the comparison window. The gain comes from parallelism: agents in this run formed three teams (architecture, schedule, and optimizer) that propose and run experiments concurrently, so multiple research directions advance within a single agent cycle, while the single-agent loop is constrained to advance one axis per experiment.

From an AUTOSCIENTISTS Champion. In the from-champion regime (Fig. 4b), both methods start from the same AUTOSCIENTISTS champion at `val_bpb` $= 0.9777$. AUTOSCIENTISTS accepts seven improvements over 93 experiments and reaches `val_bpb` $= 0.9730$. Single-agent Autoresearch accepts *zero* improvements over 100 experiments, and its best attempt reaches `val_bpb` $= 0.9783$. The seven AUTOSCIENTISTS improvements span heterogeneous research directions (query-key normalization order, matrix initialization, value-embedding gate width, final-learning-rate fraction, softcap value, compile autotuning, and a noise-floor recalibration of the starting champion) rather than concentrating on a single direction. The first improvement AUTOSCIENTISTS discovered, query-key normalization order, was never proposed by the single-agent loop in any of its 100 attempts, indicating that the gain is not just more compute but a wider set of hypotheses considered. In contrast, the single-agent loop instead repeatedly perturbed research directions already near local optima from the starting champion's tuned configuration and produced only null results.

4.4 Extending Kermut on ProteinGym Supervised Fitness Prediction

Here we evaluate whether AUTOSCIENTISTS can improve a strong existing scientific codebase rather than solve a benchmark from scratch. This setting more closely reflects how scientific research is typically conducted, where researchers usually begin from the strongest available method and iteratively search for modifications that improve it.

Setup. We use Kermut [53], a Gaussian-process method for supervised protein variant-effect prediction, as the seed program for AUTOSCIENTISTS as it is the best-performing baseline. We refer to the resulting discovered model as AUTOSCIENTISTS-Kermut. During development, AUTOSCIENTISTS is allowed to modify the Kermut implementation in any way and is evaluated on a single development assay, ACE2–Spike binding. Agents optimize the objective $\ell_{\text{eval}} = \frac{1}{3}\sum_s \rho_s$, where $s \in \{\text{random}, \text{modulo}, \text{contiguous}\}$ and ρ_s is the out-of-fold Spearman correlation under split scheme s. We choose this assay because Kermut obtains a relatively low mean Spearman's ρ, making it a challenging development target. AUTOSCIENTISTS is run for 10 cycles with no human intervention and access to one H100 GPU. One cycle corresponds to each experiment agent completing one queued experiment. After development, the discovered recipe is frozen and applied without further modification to the full ProteinGym supervised substitution benchmark of 217 DMS assays.



(a) From Autoresearch baseline

(b) From a AUTOSCIENTISTS champion

Figure 4: **AUTOSCIENTISTS sustains improvement during long-running GPT training optimization.** GPT nanochat training optimization: AUTOSCIENTISTS vs. Autoresearch [3]. **(a) From Autoresearch baseline** (`val_bpb` $= 0.998$): AUTOSCIENTISTS reaches `val_bpb` ≈ 0.978 in 34 experiments vs. 65 for Autoresearch, a $1.9\times$ speedup at the matched loss. **(b) From a AUTOSCIENTISTS champion** obtained after 50 prior AUTOSCIENTISTS experiments (`val_bpb` $= 0.9777$) with the same dead-end registry: AUTOSCIENTISTS accepts 7 out of 93 experiments to reach `val_bpb` $= 0.9730$; Autoresearch accepts 0 out of 100.

Results. On the ACE2–Spike development assay, AUTOSCIENTISTS-Kermut improves mean Spearman's ρ from 0.747 for Kermut to 0.840, a 12.5% relative improvement. The discovered predictor is a three-GP ensemble that combines Kermut's structure-kernel with expanded zero-shot features, greedy diversity-based feature selection, and quantile-warped targets. Notably, AUTOSCIENTISTS explores research directions beyond hyperparameter tuning of Kermut. Architectural details are provided in Appendix G with ablations of AUTOSCIENTISTS-introduced components in Appendix G.2. We then evaluate the frozen AUTOSCIENTISTS-Kermut recipe across all 217 supervised substitution DMS assays in ProteinGym. As shown in Table 2, AUTOSCIENTISTS-Kermut improves the official average Spearman's ρ from 0.657 for Kermut to 0.700, an absolute gain of 0.043 and a 6.5% relative improvement. The improvement in Spearman's ρ is observed across all three CV schemes. AUTOSCIENTISTS-Kermut also outperforms the other supervised baselines considered. The discovered use of quantile warping and rank-oriented model selection improve variant ordering but do not necessarily improve calibrated regression with MSE increasing slightly by 0.006. Extending AUTOSCIENTISTS to optimize multi-objective leaderboards, including MSE, is an important direction for future work.

Table 2: Performance on the ProteinGym supervised substitution benchmark, comprising 217 DMS assays across UniProt/function groups. Best results are in **bold** and second best in *italic*. Values are mean (SE).

Model type	**Model name**	**Spearman's ρ (↑)**				**MSE (↓)**			
		Contig.	Mod.	Rand.	Avg.	Contig.	Mod.	Rand.	Avg.
Embed.	ESM-1v Embeddings	0.479 (0.015)	0.514 (0.014)	0.614 (0.019)	0.535 (0.009)	0.914 (0.053)	0.848 (0.071)	0.603 (0.055)	0.789 (0.035)
	MSA Transformer Embeddings	0.513 (0.019)	0.562 (0.016)	0.670 (0.012)	0.581 (0.009)	0.860 (0.055)	0.783 (0.066)	0.529 (0.030)	0.724 (0.030)
	Tranception Embeddings	0.439 (0.014)	0.525 (0.011)	0.681 (0.015)	0.548 (0.008)	1.037 (0.048)	0.849 (0.052)	0.518 (0.043)	0.802 (0.028)
NPT	ProteinNPT	0.529 (0.018)	0.588 (0.013)	0.741 (0.015)	0.619 (0.009)	0.856 (0.051)	0.765 (0.056)	0.441 (0.046)	0.687 (0.029)
Kermut	Kermut	*0.593 (0.015)*	*0.633 (0.012)*	*0.746 (0.014)*	*0.657 (0.008)*	**0.737 (0.042)**	*0.666 (0.050)*	**0.413 (0.035)**	**0.605 (0.024)**
	AUTOSCIENTISTS	**0.635 (0.013)**	**0.681 (0.011)**	**0.783 (0.010)**	**0.700 (0.007)**	*0.812 (0.131)*	**0.607 (0.040)**	**0.413 (0.025)**	*0.611 (0.047)*

4.5 Ablations of AUTOSCIENTISTS

We next test which components of AUTOSCIENTISTS are responsible for its performance. The ablations remove one component at a time while keeping the agent backend, task interface, starting program, and experimental budget fixed. This isolates whether gains come from analyst-guided proposal generation, cross-agent feedback, team reorganization, or the shared experimental record.

Setup. To ensure the robustness and effectiveness of the key designs in AUTOSCIENTISTS, we isolate the contribution of four AUTOSCIENTISTS components on four tasks (Table 3; TDC-hERG, Cell-Cell Communication, Human Plasma-Protein Binding, and GPT nanochat training optimization), holding the agent backend, task interface, total compute, and starting program fixed. The four ablations

are: (1) **No analyst** removes the three analyst agents and reassigns their duties (proposal generation, knowledge-file maintenance, hypothesis tracking) to the experiment agents; (2) **No cross-agent feedback** disables comment threads on proposals and results, so agents cannot critique each other or share near-misses across teams; (3) **No self-organization** fixes team organization at boot and prevents agents from re-organizing teams across rounds; (4) **Independent agents** removes both cross-agent feedback and the shared state (champion program, results log, dead-end registry, and accumulated knowledge), so each agent runs a solo loop maintaining only its own private state and cannot observe any other agent's results.

Results. The full AUTOSCIENTISTS wins every task on its primary metric, and the most damaging ablation differs by task. Removing the *analyst* is most damaging on TDC-hERG (AUROC $0.867 \rightarrow 0.738$, leaderboard percentile $85.7 \rightarrow 14.3$). Removing *cross-agent feedback* is most damaging on Human Plasma-Protein Binding (Pearson correlation $0.8729 \rightarrow 0.7144$, leaderboard percentile $80 \rightarrow 30$). Removing *self-organization* is most damaging on GPT training optimization (`val_bpb` $0.9777 \rightarrow 0.9833$). The *independent-agents* cut is most damaging on Cell-Cell Communication (Odds Ratio $0.924 \rightarrow 0.435$, the largest proportional drop in the table) and no-self-organization also causes the largest degradation on GPT training. No single mechanism dominates across all four tasks and each removed component produces a AUTOSCIENTISTS that is dominated on at least one task. We read this as evidence that the four components address complementary failure modes rather than redundantly contributing the same kind of gain. Analyst-driven refinement matters when proposal quality is the bottleneck, cross-agent feedback matters when individual agents observe only a partial signal, self-organization matters when the productive search direction shifts during the run, and a shared experimental record matters when isolated agents would otherwise duplicate work or converge to incompatible local optima. Per-ablation trajectories are in Appendix D.

Table 3: Ablation results of AUTOSCIENTISTS for biomedicine and GPT training optimization tasks.

Task	Metric	No analyst	No cross-agent	No self-org.	Independent agents	AUTOSCIENTISTS
TDC-hERG	AUROC (↑)	0.738	0.819	0.807	0.853	**0.867**
Cell-Cell Communication	Odds Ratio (↑)	0.858	0.908	0.628	0.435	**0.924**
Human Plasma-Protein Binding	Pearson correlation (↑)	0.813	0.714	0.811	0.784	**0.873**
GPT Training Optimisation	Best `val_bpb` (↓)	0.9817	0.9814	0.9833	0.9833	**0.9777**

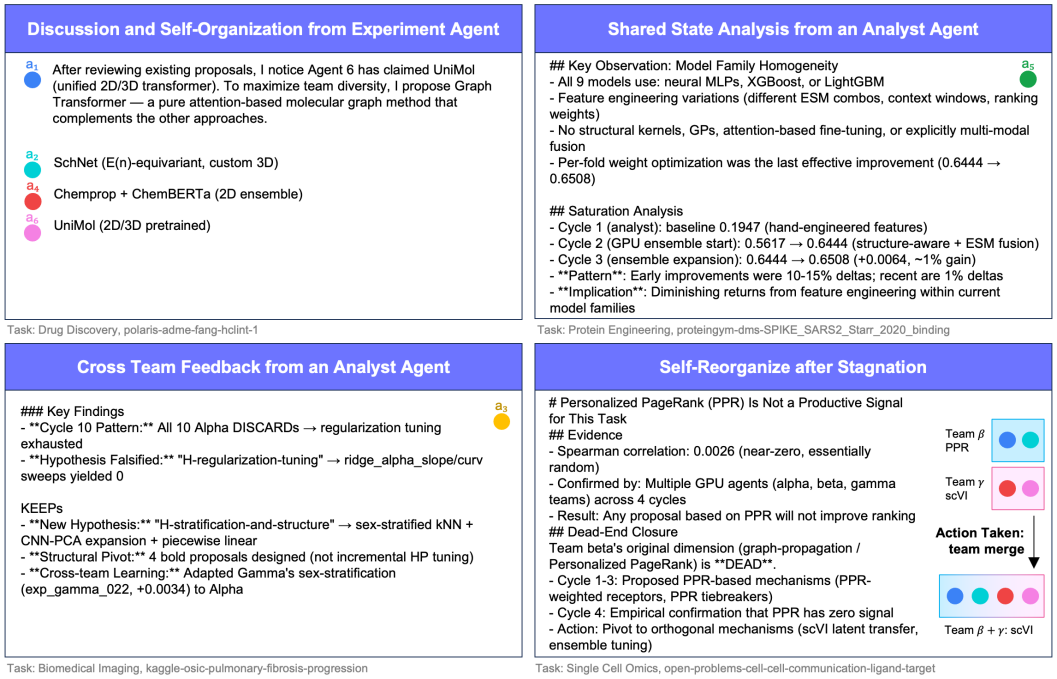


Figure 5: **Emergent coordination during long-running experimental search.** Illustrations of AUTOSCIENTISTS agent-team interactions in long-running research experiments, featuring representative quotes from the agents.

5 Limitations and Future Work

AUTOSCIENTISTS is not designed to be more LLM-call efficient than single-agent baselines. As shown in Table S8, AUTOSCIENTISTS uses more LLM tokens than Autoresearch, though within the same order of magnitude, reflecting its use of multiple agents for parallel reasoning, discussion, and team reorganization. Instead, AUTOSCIENTISTS is designed to improve experimental search under a fixed experimental-compute budget by enabling teams of agents to explore and collaborate over the design space. Under a fixed experimental-compute budget, this approach achieves better performance than existing methods as shown in Figures 3 and 4. As part of the matched experimental-compute budget used for fair comparison on BioML-Bench, we restricted AUTOSCIENTISTS to one H100 GPU per task so GPU-bound experiments in that evaluation were executed sequentially. This setting evaluates experiment selection under matched compute but does not fully exercise AUTOSCIENTISTS's capacity for parallel experimentation. When multiple GPUs are available, AUTOSCIENTISTS can dispatch experiments concurrently. Evaluating how AUTOSCIENTISTS scales with larger GPU budgets remains future work. Additionally, the number of agents is set before running. In future work, we will work towards dynamically scaling team size depending on task difficulty. We provide preliminary exploration of different AUTOSCIENTISTS team sizes in Appendix B.2.

6 Conclusion

We introduced AUTOSCIENTISTS, a self-organizing agent team for long-horizon scientific experimentation. AUTOSCIENTISTS addresses a limitation of existing AI agents, which can run individual experiments but often struggle to sustain experimental search as evidence accumulates and productive directions change. Across BioML-Bench, GPT training optimization, and ProteinGym, AUTOSCIENTISTS improves over state-of-the-art AI agents under matched experimental budgets.

AUTOSCIENTISTS makes long-running experimentation a collective search process. Agents evaluate proposals before execution, record successful and failed directions, share evidence through a common state, and reorganize teams when progress stalls. This design helps agents use experimental trials more effectively, avoid repeated dead ends, and continue identifying productive modifications after individual agents plateau.

Acknowledgments

A.F. is supported by the Kempner Graduate Fellowship at Harvard University. We gratefully acknowledge the support by NSF CAREER Award 2339524, ARPA-H Biomedical Data Fabric (BDF) Toolbox Program, Amazon Faculty Research, Google Research Scholar Program, AstraZeneca Research, GlaxoSmithKline Award, Roche Alliance with Distinguished Scientists (ROADS) Program, Sanofi iDEA-iTECH Award, Boehringer Ingelheim Award, Merck Award, Optum AI Research Collaboration Award, Pfizer Research, Gates Foundation (INV-079038), Chan Zuckerberg Initiative, Collaborative Center for XDP at Massachusetts General Hospital, John and Virginia Kaneb Fellowship at Harvard Medical School, Biswas Computational Biology Initiative in partnership with the Milken Institute, Harvard Medical School Dean's Innovation Fund for the Use of Artificial Intelligence, and the Kempner Institute for the Study of Natural and Artificial Intelligence at Harvard University. This work was delivered, in part, through the AURORA project supported by the Cancer Grand Challenges partnership funded by Cancer Research UK. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funders.

References

[1] Shanghua Gao, Ada Fang, Yepeng Huang, Valentina Giunchiglia, Ayush Noori, Jonathan Richard Schwarz, Yasha Ektefaie, Jovana Kondic, and Marinka Zitnik. Empowering biomedical discovery with ai agents. *Cell*, 187(22):6125–6151, 2024.

[2] Henry E. Miller, Matthew Greenig, Benjamin Tenmann, and Bo Wang. BioML-bench: Evaluation of AI agents for end-to-end biomedical ML. *bioRxiv*, 2025. doi: 10.1101/2025.09.01.673319. URL https://www.biorxiv.org/content/early/2025/09/28/2025.09.01.673319.

[3] Andrej Karpathy. Autoresearch: AI agents running research on single-GPU nanochat training automatically. https://github.com/karpathy/autoresearch, 2026. GitHub repository.

[4] Ludovico Mitchener, Angela Yiu, Benjamin Chang, Mathieu Bourdenx, Tyler Nadolski, Arvis Sulovari, Eric C Landsness, Daniel L Barabasi, Siddharth Narayanan, Nicky Evans, et al. Kosmos: An AI scientist for autonomous discovery. *arXiv preprint arXiv:2511.02824*, 2025.

[5] Ruofan Jin, Mingyang Xu, Fei Meng, Guancheng Wan, Qingran Cai, Yize Jiang, Jin Han, Yuanyuan Chen, Wanqing Lu, Mengyang Wang, Zhiqian Lan, Yuxuan Jiang, Junhong Liu, Dongyao Wang, Le Cong, and Zaixi Zhang. Stella: Towards a biomedical world model with self-evolving multimodal agents. *bioRxiv*, 2026. doi: 10.1101/2025.07.01.662467. URL https://www.biorxiv.org/content/early/2026/01/25/2025.07.01.662467.

[6] Shanghua Gao, Richard Zhu, Zhenglun Kong, Ayush Noori, Xiaorui Su, Curtis Ginder, Theodoros Tsiligkaridis, and Marinka Zitnik. Txagent: an ai agent for therapeutic reasoning across a universe of tools. *arXiv preprint arXiv:2503.10970*, 2025.

[7] Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, et al. Towards an ai co-scientist. *arXiv preprint arXiv:2502.18864*, 2025.

[8] José R Penadés, Juraj Gottweis, Lingchen He, Jonasz B Patkowski, Alexander Daryin, Wei-Hung Weng, Tao Tu, Anil Palepu, Artiom Myaskovsky, Annalisa Pawlosky, et al. Ai mirrors experimental science to uncover a mechanism of gene transfer crucial to bacterial evolution. *Cell*, 188(23):6654–6665, 2025.

[9] Pengwei Sui, Michelle M. Li, Shanghua Gao, Wanxiang Shen, Valentina Giunchiglia, Andrew Shen, Yepeng Huang, Zhenglun Kong, and Marinka Zitnik. Medea: An omics ai agent for therapeutic discovery. *bioRxiv*, 2026. doi: 10.64898/2026.01.16.696667. URL https://www.biorxiv.org/content/early/2026/01/20/2026.01.16.696667.

[10] Kexin Huang, Serena Zhang, Hanchen Wang, Yuanhao Qu, Yingzhou Lu, Yusuf Roohani, Ryan Li, Lin Qiu, Gavin Li, Junze Zhang, et al. Biomni: A general-purpose biomedical AI agent. *biorxiv*, 2025.

[11] Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. AIDE: AI-driven exploration in the space of code. *arXiv preprint arXiv:2502.13138*, 2025.

[12] Ao Qu, Han Zheng, Zijian Zhou, Yihao Yan, Yihong Tang, Shao Yong Ong, Fenglu Hong, Kaichen Zhou, Chonghe Jiang, Minwei Kong, et al. Coral: Towards autonomous multi-agent evolution for open-ended discovery. *arXiv preprint arXiv:2604.01658*, 2026.

[13] Shiyang Feng, Runmin Ma, Xiangchao Yan, Yue Fan, Yusong Hu, Songtao Huang, Shuaiyu Zhang, Zongsheng Cao, Tianshuo Peng, Jiakang Yuan, et al. Internagent-1.5: A unified agentic framework for long-horizon autonomous scientific discovery. *arXiv preprint arXiv:2602.08990*, 2026.

[14] Kyle Swanson, Wesley Wu, Nash L Bulaong, John E Pak, and James Zou. The virtual lab of ai agents designs new sars-cov-2 nanobodies. *Nature*, 646(8085):716–723, 2025.

[15] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first international conference on machine learning*, 2024.

[16] Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. ReConcile: Round-table conference improves reasoning via consensus among diverse LLMs, 2024. URL https://arxiv.org/abs/2309.13007.

[17] Pascal Notin, Aaron Kollasch, Daniel Ritter, Lood van Niekerk, Steffanie Paul, Han Spinner, Nathan Rollins, Ada Shaw, Rose Orenbuch, Ruben Weitzman, Jonathan Frazer, Mafalda Dias, Dinko Franceschi, Yarin Gal, and Debora Marks. Proteingym: Large-scale benchmarks for protein fitness prediction and design. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt,

and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 64331–64379. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/cac723e5ff29f65e3fcbb0739ae91bee-Paper-Datasets_and_Benchmarks.pdf.

[18] Mourad Gridach, Jay Nanavati, Khaldoun Zine El Abidine, Lenon Mendes, and Christina Mack. Agentic AI for scientific discovery: A survey of progress, challenges, and future directions, 2025. URL https://arxiv.org/abs/2503.08979.

[19] Chengwei Liu, Chong Wang, Jiayue Cao, Jingquan Ge, Kun Wang, Lyuye Zhang, Ming-Ming Cheng, Penghai Zhao, Tianlin Li, Xiaojun Jia, Xiang Li, Xingshuai Li, Yang Liu, Yebo Feng, Yihao Huang, Yijia Xu, Yuqiang Sun, Zhenhong Zhou, and Zhengzi Xu. A vision for auto research with LLM agents, 2025. URL https://arxiv.org/abs/2504.18765.

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

[21] Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi, Sergey Feldman, Dany Haddad, Jena D. Hwang, Peter Jansen, Varsha Kishore, Bodhisattwa Prasad Majumder, Aakanksha Naik, Sigal Rahamimov, Kyle Richardson, Amanpreet Singh, Harshit Surana, Aryeh Tiktinsky, Rosni Vasu, Guy Wiener, Chloe Anastasiades, Stefan Candra, Jason Dunkelberger, Dan Emery, Rob Evans, Malachi Hamada, Regan Huff, Rodney Kinney, Matt Latzke, Jaron Lochner, Ruben Lozano-Aguilera, Cecile Nguyen, Smita Rao, Amber Tanaka, Brooke Vlahos, Peter Clark, Doug Downey, Yoav Goldberg, Ashish Sabharwal, and Daniel S. Weld. AstaBench: Rigorous benchmarking of AI agents with a scientific research suite, 2025. URL https://arxiv.org/abs/2510.21652.

[22] Shuo Yan, Ruochen Li, Ziming Luo, Zimu Wang, Daoyang Li, Liqiang Jing, Kaiyu He, Peilin Wu, George Michalopoulos, Yue Zhang, Ziyang Zhang, Mian Zhang, Zhiyu Chen, and Xinya Du. LMR-BENCH: Evaluating LLM agent's ability on reproducing language modeling research, 2025. URL https://arxiv.org/abs/2506.17335.

[23] Dechao Bu, Jingbo Sun, Kun Li, Zihao He, Wei Huang, Jinlin Hu, Shanshan Zhang, Shuangshuang Lei, Peipei Huo, Zhihao Wang, et al. Empowering ai data scientists using a multi-agent llm framework with self-evolving capabilities for autonomous, tool-aware biomedical data analyses. *Nature Biomedical Engineering*, pages 1–16, 2026.

[24] 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.

[25] Haoyang Liu, Yijiang Li, and Haohan Wang. GenoMAS: A multi-agent framework for scientific discovery via code-driven gene expression analysis. *arXiv preprint arXiv:2507.21035*, 2025.

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

[27] Keyan Ding, Jing Yu, Junjie Huang, Yuchen Yang, Qiang Zhang, and Huajun Chen. Scitoolagent: a knowledge-graph-driven scientific agent for multitool integration. *Nature Computational Science*, 5(10):962–972, 2025.

[28] Dong Han, Zhehong Ai, Pengxiang Cai, Shanya Lu, Jianpeng Chen, Zihao Ye, Shuzhou Sun, Ben Gao, Lingli Ge, Weida Wang, et al. ChemBOMAS: Accelerated BO in chemistry with LLM-enhanced multi-agent system. *arXiv preprint arXiv:2509.08736*, 2025.

[29] Shijie Xia, Yuhan Sun, and Pengfei Liu. SR-scientist: Scientific equation discovery with agentic AI. *arXiv preprint arXiv:2510.11661*, 2025.

[30] Xiao Wu, Ting-Zhu Huang, Liang-Jian Deng, Xiaobing Yu, Yu Zhong, Shangqi Deng, Ufaq Khan, Jianghao Wu, Xiaofeng Liu, Imran Razzak, Xiaojun Chang, and Yutong Xie. SelfAI: A self-directed framework for long-horizon scientific discovery, 2025. URL https://arxiv.org/abs/2512.00403.

[31] Yougang Lyu, Xi Zhang, Xinhao Yi, Yuyue Zhao, Shuyu Guo, Wenxiang Hu, Jan Piotrowski, Jakub Kaliski, Jacopo Urbani, Zaiqiao Meng, Lun Zhou, and Xiaohui Yan. EvoScientist: Towards multi-agent evolving AI scientists for end-to-end scientific discovery, 2026. URL https://arxiv.org/abs/2603.08127.

[32] Xu Huang, Junwu Chen, Yuxing Fei, Zhuohan Li, Philippe Schwaller, and Gerbrand Ceder. CASCADE: Cumulative agentic skill creation through autonomous development and evolution. *arXiv preprint arXiv:2512.23880*, 2025.

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

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

[35] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Language agents as optimizable graphs. *arXiv preprint arXiv:2402.16823*, 2024.

[36] Jen tse Huang, Jiaxu Zhou, Tailin Jin, Xuhui Zhou, Zixi Chen, Wenxuan Wang, Youliang Yuan, Michael Lyu, and Maarten Sap. On the resilience of LLM-based multi-agent collaboration with faulty agents. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=bkiM54QftZ.

[37] Frédéric Berdoz, Leonardo Rugli, and Roger Wattenhofer. Can ai agents agree? *arXiv preprint arXiv:2603.01213*, 2026.

[38] Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D. Nguyen. Multi-agent collaboration mechanisms: A survey of LLMs, 2025. URL https://arxiv.org/abs/2501.06322.

[39] Yingxuan Yang, Chengrui Qu, Muning Wen, Laixi Shi, Ying Wen, Weinan Zhang, Adam Wierman, and Shangding Gu. Understanding agent scaling in LLM-based multi-agent systems via diversity. *arXiv preprint arXiv:2602.03794*, 2026.

[40] Yubin Kim, Ken Gu, Chanwoo Park, Chunjong Park, Samuel Schmidgall, A Ali Heydari, Yao Yan, Zhihan Zhang, Yuchen Zhuang, Mark Malhotra, et al. Towards a science of scaling agent systems. *arXiv preprint arXiv:2512.08296*, 2025.

[41] Aneesh Pappu, Batu El, Hancheng Cao, Carmelo di Nolfo, Yanchao Sun, Meng Cao, and James Zou. Multi-agent teams hold experts back. *arXiv preprint arXiv:2602.01011*, 2026.

[42] Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. MultiAgentBench: Evaluating the collaboration and competition of LLM agents, 2025. URL https://arxiv.org/abs/2503.01935.

[43] Jonathon N Cummings and Sara Kiesler. Collaborative research across disciplinary and organizational boundaries. *Social studies of science*, 35(5):703–722, 2005.

[44] Stefan Wuchty, Benjamin F Jones, and Brian Uzzi. The increasing dominance of teams in production of knowledge. *Science*, 316(5827):1036–1039, 2007.

[45] Fengli Xu, Lingfei Wu, and James Evans. Flat teams drive scientific innovation. *Proceedings of the National Academy of Sciences*, 119(23):e2200927119, 2022.

[46] Kara L Hall, Amanda L Vogel, Grace C Huang, Katrina J Serrano, Elise L Rice, Sophia P Tsakraklides, and Stephen M Fiore. The science of team science: A review of the empirical evidence and research gaps on collaboration in science. *American psychologist*, 73(4):532, 2018.

[47] Xinyu Zhu, Yuzhu Cai, Zexi Liu, Bingyang Zheng, Cheng Wang, Rui Ye, Jiaao Chen, Hanrui Wang, Wei-Chen Wang, Yuzhi Zhang, et al. Toward ultra-long-horizon agentic science: Cognitive accumulation for machine learning engineering. *arXiv preprint arXiv:2601.10402*, 2026.

[48] Christoph Riedl. Emergent coordination in multi-agent language models. *arXiv preprint arXiv:2510.05174*, 2025.

[49] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229, 2019.

[50] Anthropic. Claude Code: Overview. https://code.claude.com/docs/en/overview, 2026. Product documentation. Accessed: 2026-05-06.

[51] Anthropic. Claude Sonnet 4.6. https://www.anthropic.com/claude/sonnet, 2026. Model documentation. Model ID: `claude-sonnet-4-6`. Accessed: 2026-05-06.

[52] Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. Mlagentbench: Evaluating language agents on machine learning experimentation. *arXiv preprint arXiv:2310.03302*, 2023.

[53] Peter Mø rch Groth, Mads Herbert Kerrn, Lars Olsen, Jesper Salomon, and Wouter Boomsma. Kermut: Composite kernel regression for protein variant effects. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 29514–29565. Curran Associates, Inc., 2024. doi: 10.52202/079017-0929. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/34547650b2ca69d91f3b3c3ae8b21962-Paper-Conference.pdf.

[54] S. L. Lee, P. Yadav, Y. Li, J. J. Meudt, J. Strang, D. Hebel, A. Alfson, S. J. Olson, T. R. Kruser, J. B. Smilowitz, K. Borchert, B. Loritz, L. Gharzai, S. Karimpour, J. Bayouth, and M. F. Bassetti. Uw-madison gi tract image segmentation. https://kaggle.com/competitions/uw-madison-gi-tract-image-segmentation, 2022. Kaggle.

[55] Ahmed Shahin, Carmela Wegworth, David, Elizabeth Estes, Julia Elliott, Justin Zita, SimonWalsh, Slepetys, and Will Cukierski. Osic pulmonary fibrosis progression. https://kaggle.com/competitions/osic-pulmonary-fibrosis-progression, 2020. Kaggle.

[56] Will Cukierski. Histopathologic cancer detection. https://kaggle.com/competitions/histopathologic-cancer-detection, 2018. Kaggle.

[57] Adam Flanders, Chris Carr, Evan Calabrese, PhD FelipeKitamura, MD, inversion, JeffRudie, John Mongan, Julia Elliott, Luciano Prevedello, Michelle Riopel, sprint, Spyridon Bakas, and Ujjwal. Rsna-miccai brain tumor radiogenomic classification. https://kaggle.com/competitions/rsna-miccai-brain-tumor-radiogenomic-classification, 2021. Kaggle.

[58] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *arXiv preprint arXiv:2102.09548*, 2021.

[59] Polaris. Polaris: The benchmarking platform for drug discovery. https://polarishub.io/, 2026. Accessed: May 2026.

[60] Malte D Luecken, Scott Gigante, Daniel B Burkhardt, Robrecht Cannoodt, Daniel C Strobl, Nikolay S Markov, Luke Zappia, Giovanni Palla, Wesley Lewis, Daniel Dimitrov, et al. Defining and benchmarking open problems in single-cell analysis. *Nature Biotechnology*, 43(7):1035–1040, 2025.

[61] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[62] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

[63] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. Chemberta: large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020.

[64] Esther Heid, Kevin P Greenman, Yunsie Chung, Shih-Cheng Li, David E Graff, Florence H Vermeire, Haoyang Wu, William H Green, and Charles J McGill. Chemprop: a machine learning package for chemical property prediction. *Journal of chemical information and modeling*, 64 (1):9–17, 2024.

[65] RDKit: Open-source cheminformatics. https://www.rdkit.org, 2026. Accessed: May 2026.

[66] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785. URL http://doi.acm.org/10.1145/2939672.2939785.

[67] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[68] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

[69] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[70] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[71] Kevin K Yang, Niccolò Zanichelli, and Hugh Yeh. Masked inverse folding with sequence transfer for protein representation learning. *Protein Engineering, Design and Selection*, 36: gzad015, 2023.

[72] Yang Tan, Ruilin Wang, Banghao Wu, Liang Hong, and Bingxin Zhou. From high-throughput evaluation to wet-lab studies: advancing mutation effect prediction with a retrieval-enhanced model. *Bioinformatics*, 41(Supplement 1):i401–i409, 07 2025. doi: 10.1093/bioinformatics/btaf189. URL https://doi.org/10.1093/bioinformatics/btaf189.

[73] Mingchen Li, Yang Tan, Xinzhu Ma, Bozitao Zhong, Huiqun Yu, Ziyi Zhou, Wanli Ouyang, Bingxin Zhou, Pan Tan, and Liang Hong. Prosst: Protein language modeling with quantized structure and disentangled attention. *Advances in Neural Information Processing Systems*, 37: 35700–35726, 2024.

[74] Matsvei Tsishyn, Pauline Hermans, Marianne Rooman, and Fabrizio Pucci. Residue conservation and solvent accessibility are (almost) all you need for predicting mutational effects in proteins. *Bioinformatics*, 41(6):btaf322, 2025.

[75] Mustafa Tekpinar, Laurent David, Thomas Henry, and Alessandra Carbone. Prescott: a population aware, epistatic, and structural model accurately predicts missense effects. *Genome Biology*, 26(1):113, 2025.

[76] Bo Chen, Xingyi Cheng, Pan Li, Yangli-ao Geng, Jing Gong, Shen Li, Zhilei Bei, Xu Tan, Boyan Wang, Xin Zeng, et al. xtrimopglm: unified 100-billion-parameter pretrained transformer for deciphering the language of proteins. *Nature Methods*, 22(5):1028–1039, 2025.

[77] Jin Su, Chenchen Han, Yuyang Zhou, Junjie Shan, Xibin Zhou, and Fajie Yuan. Saprot: Protein language modeling with structure-aware vocabulary. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=6MRm3G4NiU.

[78] Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. In *International conference on machine learning*, pages 8946–8970. PMLR, 2022.

[79] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019.

[80] Zuobai Zhang, Pascal Notin, Yining Huang, Aurélie Lozano, Vijil Chenthamarakshan, Debora Marks, Payel Das, and Jian Tang. Multi-scale representation learning for protein fitness prediction. *Advances in Neural Information Processing Systems*, 37:101456–101473, 2024.

[81] Roshan M Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. In *International conference on machine learning*, pages 8844–8856. PMLR, 2021.

[82] Céline Marquet, Julius Schlensok, Marina Abakarova, Burkhard Rost, and Elodie Laine. Expert-guided protein language models enable accurate and blazingly fast fitness prediction. *Bioinformatics*, 40(11):btae621, 11 2024. ISSN 1367-4811. doi: 10.1093/bioinformatics/btae621. URL https://doi.org/10.1093/bioinformatics/btae621.

[83] Sebastian Prillo, Wilson Wu, and Yun S Song. Ultrafast classical phylogenetic method beats large protein language models on variant effect prediction. *Advances in neural information processing systems*, 37:130265–130290, 2024.

Reproducibility Statement

We provide the source code of AUTOSCIENTISTS in https://github.com/mims-harvard/AutoScientists, which includes the launch scripts to run all experiments outlined. We use the official GitHub repositories for BioML-Bench (https://github.com/science-machine/biomlbench), ProteinGym (https://github.com/OATML-Markslab/ProteinGym), Autoresearch (https://github.com/karpathy/autoresearch), Biomni (https://github.com/snap-stanford/biomni) which we use with the base-LLM Claude Sonnet 4.6, and Kermut (https://github.com/petergroth/kermut). Further details are available in Table S1. We use only publicly available data in benchmarking from BioML-Bench, Kaggle (https://www.kaggle.com), TDC (https://tdcommons.ai), Polaris (https://polarishub.io), Open Problems (https://openproblems.bio), and ProteinGym (https://proteingym.org). For Autoresearch and AUTOSCIENTISTS we launch experiments with new Claude Code sessions with the prompt "Read the program file and start the experimental loop, do not stop until the specified wall clock time is up", and the AI agents work with no human intervention for the duration of the specified wall clock time.

Impact Statement

AUTOSCIENTISTS may accelerate machine learning and AI for Science by helping researchers explore modeling choices, use experimental compute more efficiently, and document both successful and failed hypotheses through shared logs, dead-end registries, model cards, and research reports. These benefits are most relevant for computational biomedical and scientific ML settings where experimental iteration is costly. The main risks are over-trusting automatically discovered models, overfitting to benchmark feedback, and amplifying erroneous hypotheses if validation is weak. This is especially important in biomedical applications, where outputs should not be treated as clinically or biologically actionable without expert review and independent validation.

A Implementation Details and Algorithmic Protocols

This section gives the implementation behind the method in Section 3. A AUTOSCIENTISTS run proceeds in three phases. *Launch* (App. A.1): the system provisions a roster of experiment agents and analysts with no team assignments and posts a single discussion trigger to the forum $\mathcal{F}$. *Self-organized team formation* (App. A.3): all agents enter a structured discussion in which they propose research directions, vote on proposals, and write the team roster R; the same protocol re-opens whenever an analyst detects stagnation. *Normal cycle*: once R exists, agents are repeatedly invoked through a fixed rotation. On each invocation, an analyst generates new proposals for the team queue via coverage audit, empirical-priors ranking, and diversity rules (App. A.7), and an experiment agent claims a queued experiment, runs it, and applies a noise-aware promotion gate to decide whether to update the champion (App. A.6). All coordination flows through the shared state $\mathcal{S}$, which agents access through a list–decide–read protocol (App. A.4) and which contains per-team queues Q_k governed by optimistic locking (App. A.5). Algorithms 1–4 give the per-invocation pseudocode and are discussed in App. A.2.

A.1 Setup and Launch

A AUTOSCIENTISTS deployment is initialized by a deterministic bootstrap procedure that performs three steps. First, the shared state $\mathcal{S}$ is initialized with the task specification and an empty experiment log. Second, the agent roster is created: experiment agents (each bound to one GPU if needed) and analysts; each agent is issued a unique credential and a role-specific heartbeat description that determines its behavior at every invocation. The agents are launched with no team assignments: the roster R that maps agents to teams is initially empty. Third, a single `[DISCUSSION-TRIGGER]` is posted to the forum $\mathcal{F}$, initiating the bootstrap discussion in which agents propose research directions and self-organize into teams (Section 3, App. A.3).

After bootstrap, agents are repeatedly invoked in a fixed rotation by a simple loop. Each invocation is a single LLM session that wakes up, executes one heartbeat, and exits; long-horizon coordination is achieved by repeating these invocations. The loop passes only the agent's identity, and the agent reads its persistent state (credentials, role assignment, and GPU binding for experiment agents) and the shared state $\mathcal{S}$ on every invocation to discover team membership, the current champion p^*, the

Table S1: Existing assets used in this work. We report the asset type, how it was used, citation, version or commit when available, license, and URL or terms of use.

Asset	Type	Use in this paper	Citation	Version / commit	License	URL
BioML-Bench	Benchmark / datasets / code	Evaluation on 24 biomedical ML tasks.	[2]	`673f3d8`	MIT	link
ProteinGym	Benchmark / datasets / code	Protein engineering tasks and 217-assay benchmark.	[17]	PG_v1.3	MIT	link
Autoresearch	Codebase / baseline	Single-agent baseline and GPT nanochat comparison.	[3]	`228791f`	MIT	link
Biomni	Codebase / baseline	Biomedical agent baseline on BioML-Bench.	[10]	0.0.8	Apache-2.0	link
Kermut	Codebase / baseline model	Seed method for AUTOSCIENTISTS-Kermut.	[53]	`7e9e2e6`	MIT	link
Claude Code / Claude Sonnet 4.6	LLM / coding agent	Base coding-agent backend.	[50, 51]	Claude Sonnet 4.6	API terms	link
Kaggle UW-Madison GI tract	Dataset	Biomedical imaging benchmark task.	[54]	Jul. 2022	Non-commercial academic research	link
Kaggle OSIC pulmonary fibrosis	Dataset	Biomedical imaging benchmark task.	[55]	Oct. 2020	Non-commercial academic research	link
Kaggle histopathologic cancer detection	Dataset	Biomedical imaging benchmark task.	[56]	Mar. 2019	Non-commercial academic research	link
Kaggle RSNA-MICCAI brain tumor	Dataset	Biomedical imaging benchmark task.	[57]	Oct. 2021	Non-commercial academic research	link
TDCommons	Dataset	Drug discovery benchmark tasks.	[58]	`c310c35`	MIT	link
Polaris Hub tasks	Dataset	Drug discovery benchmark tasks.	[59]	0.13.0	Apache-2.0	link
Open Problems	Dataset / benchmark	Single-cell omics benchmark tasks.	[60]	`5d53ffb`	MIT	link
ESM-2 models	Pretrained model	Embeddings and fine-tuning components.	[61]	650M, 3B	MIT	link
ProteinMPNN	Pretrained model / features	Conditional-probability and structure-kernel features	[62]	`8907e66`	MIT	link
ChemBERTa	Pretrained model	Molecular embeddings.	[63]	Zinc-Base-V1	MIT	link
Chemprop	Codebase / model	Models for BioML-Bench.	[64]	v2.2.3	MIT	link
RDKit	Software library	Molecular fingerprints and descriptors.	[65]	Q1 2026	BSD-3-Clause	link
XGBoost	Software library	Gradient-boosted tree models.	[66]	3.2.0	Apache-2.0	link
LightGBM	Software library	Gradient-boosted tree models.	[67]	4.6.0	MIT	link
CatBoost	Software library	Gradient-boosted tree models.	[68]	1.2.10	Apache-2.0	link
EfficientNet	Pretrained model	Embeddings and fine-tuning components.	[69]	B0, B3, B4	Apache-2.0	link
PyTorch / PyTorch Geometric	Software libraries	Neural network training and implementation.	[70]	2.11.0	BSD-3-Clause	link

queue state, and recent forum activity. Re-discussion rounds are triggered by agents themselves when stagnation is detected (Appendix A.2).

Algorithm 1 Heartbeat dispatch (per agent invocation). The discussion branch on line 3 is detailed in Algorithm 2; the role-specific normal cycle on line 9 is detailed in Algorithm 4 (analysts) or Algorithm 3 (experiment agents).

Require: agent identity i

1: Read roster R and recent forum $\mathcal{F}$ from $\mathcal{S}$
2: **if** $\mathcal{F}$ has an unresolved `[DISCUSSION-TRIGGER]` **or** R is empty **then**
3: **run** Algorithm 2; **return** ▷ discussion
4: **else if** i is not assigned to any team in R **then**
5: **return** ▷ no-team exit
6: **else if** i is an experiment agent **and** an unposted result exists **then**
7: post pending `[RESULT]`; **return** ▷ resume
8: **end if**
9: Read $\mathcal{T}_k$, p^*, Q_k, knowledge files ▷ App. A.4
10: **run** Algorithm 4 if i is an analyst, **else** Algorithm 3 ▷ normal cycle
11: persist updated state and exit

Algorithm 2 Self-organized team formation (discussion branch). Detailed in App. A.3.

1: Read task specification, current champion p^* if any, and the active `[DISCUSSION-TRIGGER]` thread from $\mathcal{F}$
2: post candidate research directions to $\mathcal{F}$
3: post ranked hypotheses about the proposed directions, with reasoning
4: critique earlier posts and identify gaps in the proposed decomposition
5: post a self-termination vote on the trigger thread: `[DISCUSS-MORE]` or `[DISCUSS-DONE]`
6: **if** a majority of agents have cast `[DISCUSS-DONE]` on the trigger thread **and** i is the alphabetically-last analyst that participated **then**
7: consolidate proposals into a roster $R = \{(\mathcal{T}_k, \text{axis}_k, \text{members}_k)\}_{k=1}^{K}$
8: write R to $\mathcal{S}$
9: **end if**

Algorithm 3 Experiment-agent cycle. Queue protocol in App. A.5; noise-aware gate in App. A.6.

1: claim a queued experiment q from Q_k; **return** if none
2: apply diff to p^*, train candidate p', compute $\Delta = \ell(p') - \ell(p^*)$
3: **if** $\Delta > M\sigma$ **then**
4: promote p' to p^*
5: **else if** $0 < \Delta \leq M\sigma$ **then**
6: re-run on a second seed; promote iff both runs strictly improve
7: **else**
8: discard
9: **end if**
10: record outcome to log $\mathcal{L}$, release claim on Q_k, post `[RESULT]` to $\mathcal{F}$

A.2 Heartbeat Protocol

Algorithm 1 specifies the per-invocation dispatch; Algorithm 2 specifies what an agent does in the discussion branch; and Algorithms 4 and 3 specify the role-specific normal cycle. After reading the current roster and recent forum activity, the agent enters one of four branches: a discussion branch when an analyst has posted a `[DISCUSSION-TRIGGER]` in response to stagnation (Algorithm 4) or when the roster has not yet been formed during cold-start bootstrap; a no-team exit when the agent is not assigned to any team; a resume branch when an experiment agent has an unposted result from a prior session; or the role-specific normal cycle. In the normal cycle, the two roles divide labor: an experiment agent claims a queued experiment, applies the code change, trains, gates the result against

Algorithm 4 Analyst cycle (when i is an analyst). Each step detailed in App. A.7.

if recent experiments produced no improvement, or proposals have concentrated on a narrow class of changes **then**
    post `[DISCUSSION-TRIGGER]` to $\mathcal{F}$ ▷ App. A.3
end if
audit untested parameters of p^*
compute axis priors $\mu_{a,d}$ from experiment log $\mathcal{L}$
propose 2 experiments under ambition quota and diversity rules; if p^* changed since the analyst's last cycle, ≥ 1 proposal must target the property responsible
append proposals to Q_k ▷ App. A.5

the noise floor, and records the outcome (Algorithm 3); an analyst, rather than running experiments, maintains the team's knowledge by auditing untested parameters, ranking research directions by empirical effect size, and queueing new proposals for experiment agents to execute (Algorithm 4). In the discussion branch (Algorithm 2), both roles contribute research directions, hypotheses, and a self-termination vote; the alphabetically-last analyst that participated in the discussion is additionally responsible for consolidating the proposals into the new roster R (App. A.3). After the branch returns, the agent persists its updated state and exits.

A.3 Self-Organized Team Formation

The roster $R = \{(\mathcal{T}_k, \text{axis}_k, \text{members}_k)\}_{k=1}^{K}$ is not specified by an external coordinator; agents write it themselves through a structured discussion that opens whenever the discussion branch of Algorithm 1 is entered. There are two such situations, which follow the same protocol: *cold-start bootstrap* at the beginning of a run, when R is empty and every agent is routed to the discussion branch; and *mid-run reformation*, when an analyst's stagnation check (Algorithm 4) posts a fresh `[DISCUSSION-TRIGGER]` to $\mathcal{F}$ and subsequent agents observe it.

Discussion contributions. Each agent entering the discussion branch reads the task specification, the current champion p^* if any, and the active `[DISCUSSION-TRIGGER]` thread. It then posts (i) candidate research directions, (ii) hypotheses ranking those research directions by expected effect with the reasoning that justifies the ranking, and (iii) a self-termination vote on the trigger thread, either `[DISCUSS-MORE]` (more discussion is needed before a roster can be written) or `[DISCUSS-DONE]` (the discussion has converged). Later agents critique earlier posts, identify gaps, propose alternative axis groupings, and add their own vote.

Termination and roster writing. The discussion terminates when a majority of agents have cast `[DISCUSS-DONE]` on the trigger thread. The alphabetically-last analyst that participated in the discussion is responsible for consolidating the proposals into the new roster R and writing it to $\mathcal{S}$. The alphabetical convention is purely a tie-breaker that ensures exactly one agent assumes the consolidator role without further coordination; the system has no privileged consolidator agent.

Reformation outcomes. A new roster may create, merge, split, retire, or rebalance teams relative to the previous roster, depending on the discussion's conclusions. Teams whose research directions have produced no recent improvements can be retired; newly discovered productive research directions can spawn new teams; agents can be reassigned across teams. The persistence of R in $\mathcal{S}$ ensures that subsequent invocations begin from the freshly written decomposition.

A.4 File Discovery

Each heartbeat cycle, agents access the shared state $\mathcal{S}$ through a list–decide–read protocol: the agent first retrieves only the lightweight metadata for items in $\mathcal{S}$ (path, version, timestamp, author), then selects the items relevant to its current task, and finally fetches the contents of those items. This indirection allows new artifact types to appear in $\mathcal{S}$ over the course of a run (for instance, a newly created knowledge file or dead-end registry) without changes to the access protocol.

A.5 Queue and Claim Protocol

The team queue Q_k is a structured record of pending experiments (each with an identifier, priority, diff description, proposing agent, and link to its proposal post) and the experiments currently claimed by agents. Concurrent reads and writes are serialized by optimistic locking on a version token: a write that arrives stale is rejected and the agent retries against the latest version, so each queue update is atomic and no experiment is claimed twice. Each experiment agent releases its own claim immediately after recording the experiment's outcome.

A.6 Noise-Aware Champion Validation

The evaluation metric ℓ is stochastic: running the same program with two different random seeds yields slightly different values. A naive rule that promotes whenever $\ell(p') > \ell(p^*)$ would therefore promote candidates whose "improvement" is just noise. Because all agents build on the shared champion p^*, an erroneous promotion has compounding effects: every downstream comparison is then made against a corrupted baseline. We term this *champion pollution* and prevent it by gating every proposed promotion against an empirically measured noise floor.

For a candidate p' produced by an experiment agent (Algorithm 3), let $\Delta = \ell(p') - \ell(p^*)$ denote its signed improvement over the current champion. Since ℓ is oriented so that higher is better (Section 3), $\Delta > 0$ iff p' is strictly better. Let σ denote the per-run standard deviation of ℓ at fixed code (the *noise floor*; calibration described below), and let $M = 2$, so that $[0, M\sigma]$ is the band of improvements small enough to be confounded with noise. The gate decides

$$\text{promote}(p') = \begin{cases} \texttt{true} & \text{if } \Delta > M\sigma \\ \text{confirm}(p', \text{seed}_2) & \text{if } 0 < \Delta \le M\sigma \\ \texttt{false} & \text{if } \Delta \le 0, \end{cases} \tag{1}$$

where $\text{confirm}(p', \text{seed}_2)$ re-runs p' with a new random seed and returns `true` only if both runs strictly improve over p^*. Improvements clearly above the noise band are accepted directly; improvements inside the noise band are accepted only if a second-seed re-run confirms; non-improvements are rejected. Candidates that fall in the noise band but fail confirmation are recorded as *near-misses*.

The noise floor σ is calibrated lazily, without spending experiments specifically for calibration. Until σ has been estimated, the gate uses a conservative default noise band so that no candidate can be promoted on weak evidence. Each time the gate's middle branch fires, the resulting pair of duplicate-seed measurements $(\ell_{1,i}, \ell_{2,i})$ for the same candidate code is recorded. Once at least three such pairs have accumulated ($n \ge 3$), σ is set to the within-pair pooled standard deviation

$$\sigma = \sqrt{\frac{1}{2n} \sum_{i=1}^{n} (\ell_{1,i} - \ell_{2,i})^2}, \tag{2}$$

which estimates the per-seed noise of ℓ at fixed code without conflating it with code-to-code variation. To prevent later pairs from retroactively reclassifying earlier promotion decisions, σ is locked once five pairs have been recorded.

A.7 Analyst Proposal Protocol

On each invocation, an analyst generates two new experiment proposals for the team queue through the following cycle.

The analyst first builds a working picture of what has been tried and what has not. It scans the champion code p^* for all numeric parameters (top-level constants, class fields, and inline numeric literals) and matches each against the experiment log $\mathcal{L}$ to identify parameters that have never been varied (*baseline coverage audit*). It then computes the empirical mean effect size per (axis, direction) pair from prior experiments,

$$\mu_{a,d} = \frac{1}{|E_{a,d}|} \sum_{e \in E_{a,d}} |\Delta_e|, \tag{3}$$

designates research directions with $|E_{a,d}| < 3$ as *cold* and gives them an exploration bonus, and deprioritizes directions whose effect size falls below the noise floor ($\mu_{a,d} < \sigma$); the team queue Q_k is sorted by the resulting ranking (*empirical axis priors*).

The analyst then drafts two new proposals subject to two filters. The *ambition quota* requires at least one of the two to satisfy a bold-move criterion: a parameter-count change of at least 10%, a fix to a confirmed bug in the champion code, an experiment on an axis flagged as untested in two or more prior discussion threads, or a hypothesis-tension probe whose outcome will clearly confirm or falsify the team's current hypothesis. If neither proposal qualifies, the analyst must post a public `[EXEMPT]` comment justifying why no bold candidate exists, rather than silently accept incrementalism. The *diversity constraints* additionally require that (i) the two proposals target different research directions, (ii) no run of three or more recent same-axis proposals all push the same direction, and (iii) no proposal falls inside a previously rejected range in the dead-end registry $\mathcal{D}_k$ unless the proposer explicitly states what differs from the prior failure.

When the champion has been updated since the analyst's previous cycle (a *KEEP*: a successful promotion of a candidate to a new p^*), a final *post-KEEP* step requires at least one of the two proposals to follow up on the property responsible for the recent improvement via a different mechanism. The analyst must answer two questions to enforce this: which property of the successful change made it work, and what other untried changes share that property? The completed proposals are appended to Q_k.

B Extended Ablation Results

The main-paper ablation table (Table 3) reports the three single-component removals (`No analyst`, `No cross-agent`, `No self-org.`) against the full AUTOSCIENTISTS system. We report two additional comparisons on the same task suite here: (i) AUTOSCIENTISTS versus autoresearch [3] as a non-multi-agent reference (Table S2 for GPT training optimization and Table S7 for BioML-Bench), and (ii) a team-size sweep with working-agent counts $n \in \{2, 4, 14\}$ against the default crew ($n = 9$) (Table S3).

Table S2: **AUTOSCIENTISTS versus Autoresearch** [3] on GPT Training Optimization across two settings.

Setting	**Metric**	**Start** `val_bpb`	**# Experiments**	**Autoresearch**	**AUTOSCIENTISTS**
From Autoresearch baseline	Best `val_bpb` (↓)	0.998	50	0.9790	0.9777
From AUTOSCIENTISTS champion	Best `val_bpb` (↓)	0.9777	100	0.9777 (0 KEEPs)	**0.9730** (7 KEEPs)

B.1 Comparison Against Autoresearch at Matched and Extended Compute

Table S2 reports two settings. (i) *From the Autoresearch baseline*: both systems start from the original GPT training setup released with Autoresearch [3] (the unmodified `nanochat` code at `val_bpb` $= 0.998$) and run 50 experiments; AUTOSCIENTISTS reaches 0.9777, Autoresearch reaches 0.9790. (ii) *From the same* AUTOSCIENTISTS *champion* at `val_bpb` $= 0.9777$ (the result of AUTOSCIENTISTS's setting (i) run) for 100 additional experiments: AUTOSCIENTISTS accepts seven KEEPs and reaches 0.9730, while Autoresearch accepts zero KEEPs and produces no improvement over the starting champion. AUTOSCIENTISTS's advantage over the single-agent loop is not a constant offset: it grows with compute as the single-agent loop saturates while AUTOSCIENTISTS continues to discover new productive directions.

Table S3: **AUTOSCIENTISTS agent team size sweep.** The same five tasks as Table 3, comparing the default AUTOSCIENTISTS crew ($n = 9$) against three crew-size variants ($n = 2$, $n = 4$, $n = 14$). n counts working agents (experiment + analyst) that fire heartbeats during the search loop; the admin agent only fires once at bootstrap and is excluded. Bold indicates the best result per row.

		AUTOSCIENTISTS agent team size			
Task	**Metric**	**n=2**	**n=4**	**n=9 (default)**	**n=14**
TDC-hERG	AUROC (↑)	0.780	0.803	**0.867**	0.843
	Leaderboard % (↑)	14.3	14.3	**85.7**	42.9
ProteinGym SPIKE-SARS2	Spearman's ρ (↑)	**0.874**	0.835	0.670	0.506
	Leaderboard % (↑)	**100.0**	**100.0**	81.8	72.7
GPT Training Optimisation	Best `val_bpb` (↓)	**0.9777**	0.9778	**0.9777**	0.9821

B.2 Team-Size Sensitivity: Parallelism Gain and Oversubscription

We compare four working crew sizes (n=2, 4, 9, 14) on three tasks (TDC-hERG, ProteinGym SPIKE-SARS2, GPT nanochat training optimization). Crew composition (experiment agents + analysts): n=2 (1+1), n=4 (2+2), n=9 (6+3, the default AUTOSCIENTISTS configuration), n=14 (9+5).

Crew-size sensitivity is task-dependent. The three tasks in Table S3 show different sensitivities to crew size, and the crew size that achieves the top score differs across tasks. On TDC-hERG, AUROC spans 0.780–0.867 and leaderboard percentile spans 14.3–85.7 across the four crews. On ProteinGym SPIKE-SARS2, Spearman ρ spans 0.506–0.874 and leaderboard percentile spans 72.7–100.0. On GPT training optimization, `val_bpb` spans 0.9777–0.9821. At n=14, every task degrades relative to its best score in the table. These results suggest that the optimal crew size is task-dependent rather than a fixed property of the protocol. Dynamically scaling team size to task difficulty is a direction for future work.

Parallel execution: more agents run faster at similar quality. AUTOSCIENTISTS agents are designed to run in parallel: within each heartbeat all n working agents fire one LLM call simultaneously. Due to resource limits in our setup, we executed the agents iteratively rather than fully in parallel, so we report two complementary views. The per-experiment axis (Fig. S1) compares final quality under matched experimental compute: the three smaller crews converge to essentially the same final `val_bpb` ($\sim$ 0.9777, within the noise floor) over 51, 38, and 71 experiments for n=2, 4, 9 respectively. The per-heartbeat axis (Fig. S2) projects the wall-clock cost of the same runs under fully parallel execution: $\approx$ 26, $\approx$ 10, and $\approx$ 8 heartbeats for n=2, 4, 9, a $\sim 3.25\times$ speed-up of n=9 over n=2. So while per-experiment quality is similar across crews in the productive range, parallel execution lets larger crews finish substantially faster.

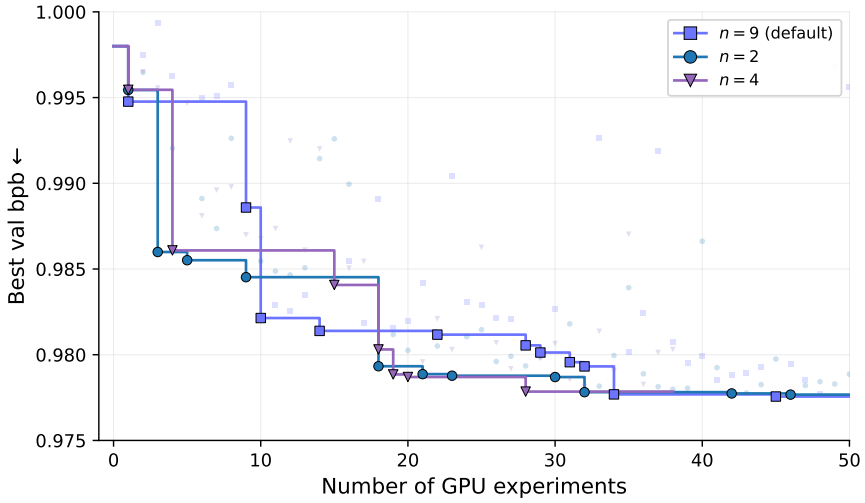


Figure S1: Team-size sweep on GPT nanochat as a function of *experiment number*. Solid step curves are the running best-so-far validation `val_bpb`; opaque markers indicate accepted KEEPs; transparent markers show per-experiment attempts that did not improve the running min. Final values: n=2 0.9777 (11 KEEPs / 51 exps), n=4 0.9778 (7 KEEPs / 38 exps), default AUTOSCIENTISTS (n=9) 0.9777 (11 KEEPs / 71 exps), n=14 0.9821 (15 KEEPs / 50 exps).

C Run-to-Run Stability

We executed three independent cold-start runs of AUTOSCIENTISTS on the GPT nanochat task to assess whether AUTOSCIENTISTS's behavior is stable across the stochastic factors that arise during cold-start coordination (workshop-post acquisition order, concurrent claim races, per-experiment random seeds). The three runs use identical hardware, templates, and task specification; none uses a fixed seed or deterministic-replay mechanism, so each constitutes an independent realization.

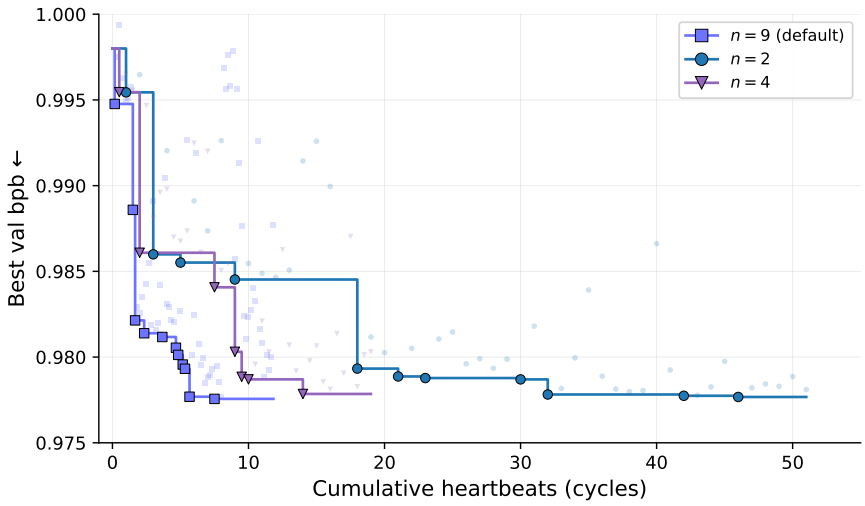


Figure S2: Same trajectories as Fig. S1, plotted against *cumulative heartbeats* under fully parallel execution (one heartbeat = one synchronized cycle in which all n working agents fire one LLM call simultaneously). n=2 finishes in ≈26 heartbeats; n=4 in ≈10; default n=9 in ≈8; n=14 in ≈4. Larger crews finish faster under parallel execution.

Figure S3 reports the running best-so-far `val_bpb` for all three runs. All three runs converge to the same best-so-far region, with final values of 0.9777 (Run 1), 0.9795 (Run 2), and 0.9780 (Run 3) after 75, 64, and 62 experiments respectively. The mean final `val_bpb` across runs is 0.9784 with sample standard deviation 0.0010 (sample variance 9.4×10^{-7}, range 0.0018). Acceptance rates are comparable across runs: $8/75 = 10.7\%$ for Run 1, $7/64 = 10.9\%$ for Run 2, and $6/62 = 9.7\%$ for Run 3. The runs select different intermediate research directions (Newton–Schulz iteration count and short-window patterns in Run 1; final-LR-fraction and per-group learning-rate allocation in Run 2; total-batch-size halving and rotary-base scaling in Run 3) yet reach the 0.978–0.980 region, indicating that the productive search surface is reached robustly under different proposal orderings rather than relying on a single deterministic path. Descent timing differs across runs because each run is gated by a small number of high-leverage probes whose discovery order is stochastic: Run 3 trails Runs 1–2 through the middle range and recovers at experiment 37 once a coupling-preserving warmdown-ratio probe lands, then makes the largest single jump of any run (a width-up KEEP at experiment 43 worth −0.0048 `val_bpb`) and reaches its final region within five experiments. The 0.0018 gap between Runs 2 and 1 reflects a single `depth_7` schedule probe that Run 1 reached at experiment 54 and Run 2 was halted before reaching; Run 3 closes most of the same gap through an alternate path.

D Per-Experiment Trajectories for Ablation Runs

This appendix plots per-experiment trajectories for the ablation runs reported in Table 3. All ablations start from the pristine nanochat baseline (`val_bpb` = 0.998) and report the running best-so-far validation `val_bpb`; transparent markers show every attempted experiment, and opaque markers indicate accepted improvements (KEEPs). Autoresearch [3] and the full AUTOSCIENTISTS system on the same anchor are included in each plot for reference.

D.1 No Self-Organization (`abl-no-self-org`)

Removing self-organization (teams are fixed at boot, with no mid-run reformation) reduces the system to 5 KEEPs and a final `val_bpb` of 0.9833 over 47 unique experiments (Fig. S4). The full AUTOSCIENTISTS system on the same anchor reaches 11 KEEPs and 0.9777 in 71 experiments. The ablated system tracks the full system closely for the first ∼10 experiments (when initial high-effect-size research directions are still being swept) but stalls thereafter, consistent with the `ISSUES.md` signals that hypothesis falsification triggers fired (analyst1 on arch v3, analyst2 on optim v3) but

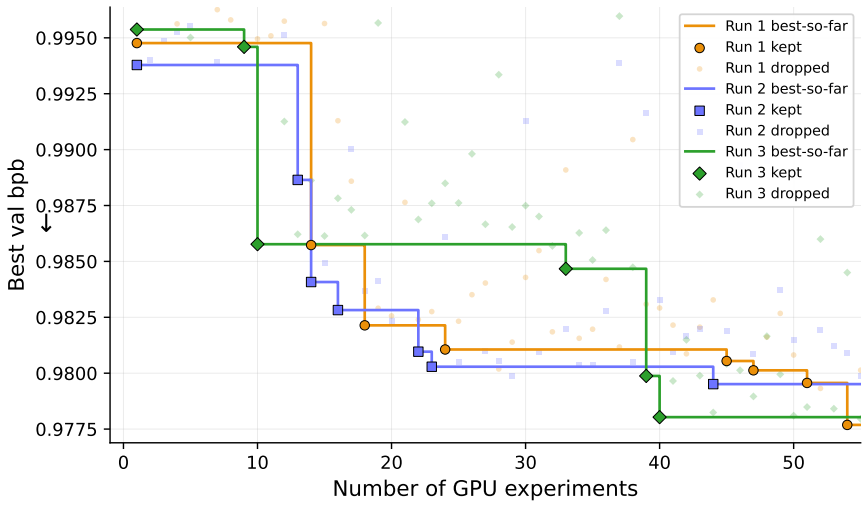


Figure S3: Per-experiment trajectories of three independent cold-start runs of AUTOSCIENTISTS on the GPT nanochat task. Solid step curves show the running best-so-far validation `val_bpb`; opaque markers indicate accepted KEEPs and transparent markers show per-experiment attempts that did not improve the running minimum. All three runs converge to the same best-so-far region under different proposal orderings, with descent timing varying based on when high-leverage probes are discovered.

had no enactment mechanism with team reformation removed. Note that this run was stopped at 47 experiments per a mid-run budget revision (the original HANDOVER target was 100); we leave investigation of whether further experiments would have closed the gap to subsequent ablation runs.

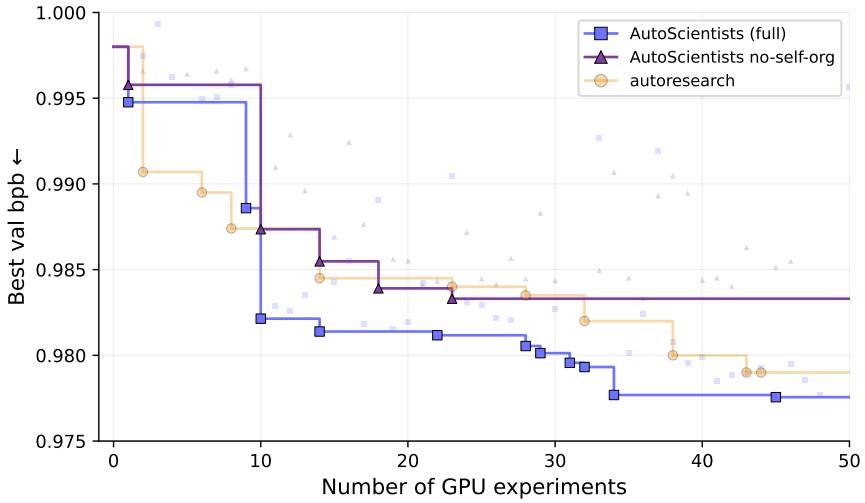


Figure S4: Per-experiment trajectory for the `abl-no-self-org` ablation compared with the full AUTOSCIENTISTS system and single-agent autoresearch, all anchored at the pristine nanochat baseline (`val_bpb` = 0.998). Solid step curves are the running best-so-far validation `val_bpb`; opaque markers indicate accepted KEEPs; transparent markers show per-experiment attempts that did not improve the running min. Final values: autoresearch 0.9773 (15 KEEPs / 83 exps), full AUTOSCIENTISTS 0.9777 (11 KEEPs / 71 exps), `abl-no-self-org` 0.9833 (5 KEEPs / 47 exps).

D.2 No Cross-Agent Communication (`abl-no-cross-agent`)

Removing cross-agent communication (only `[PROPOSAL]` and `[RESULT]` posts are allowed; all other forum activity—comments, notifications, gap analyses, rankings, synthesis posts, near-miss reports, suggestions, and structural-change proposals—is disabled, as is the stagnation-triggered re-discussion mechanism) leaves coordination only through the shared artifacts: champion record, dead-end registry, team queue, and result log. The resulting system reaches 9 KEEPs and a final `val_bpb` of 0.9814 over 50 unique experiments (Fig. S5), against 11 KEEPs and 0.9777 in 71 experiments for the full system on the same anchor. Unlike the no-self-organization ablation, the no-cross-agent run continues to find improvements throughout the budget — the artifact surface alone carries enough signal for individual axis decisions to converge to a similar neighborhood. The cost is throughput: the run needed roughly $1.85\times$ more experiments than a comparable full-system run (the run's headline result compares against an internal pristine run that reached 0.980289 in 27 experiments) and never bridged the last $\sim$0.001 gap, because teams duplicated brackets that cross-team comments and team-restructuring proposals would otherwise have deduplicated. The single largest individual win in this run was `sched_batch_half` (TOTAL_BATCH_SIZE $2^{19} \to 2^{18}$, $\Delta = -0.0067$), which agents discovered independently after several teams converged on a step-limited diagnosis from the artifact log alone.

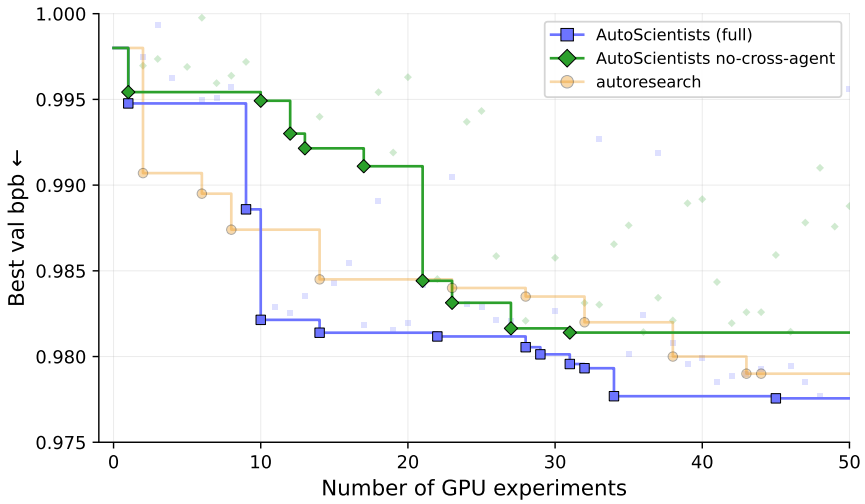


Figure S5: Per-experiment trajectory for the `abl-no-cross-agent` ablation compared with the full AUTOSCIENTISTS system and single-agent autoresearch, all anchored at the pristine nanochat baseline (`val_bpb` $= 0.998$). Solid step curves are the running best-so-far validation `val_bpb`; opaque markers indicate accepted KEEPs; transparent markers show per-experiment attempts that did not improve the running min. Final values: autoresearch 0.9773 (15 KEEPs / 83 exps), full AUTOSCIENTISTS 0.9777 (11 KEEPs / 71 exps), `abl-no-cross-agent` 0.9814 (9 KEEPs / 50 exps).

D.3 No Analyst Role (`abl-no-analyst`)

Removing the analyst role (no analyst agents at all; experiment agents handle proposal generation, axis prioritization, and post-KEEP induction in addition to executing experiments) leaves the system with five experiment agents operating against the shared workshop and artifact surface alone. The resulting system reaches 7 KEEPs and a final `val_bpb` of 0.9817 over 50 unique experiments (Fig. S6), against 11 KEEPs and 0.9777 in 71 experiments for the full system on the same anchor. The most surprising finding is that experiment agents successfully self-organized cross-team probes when their team's native research directions were exhausted: after the schedule team closed its TOTAL_BATCH / WARMUP / WARMDOWN / FINAL_LR_FRAC bracket, gpu3 re-bracketed the architecture team's MLP_RATIO under the new compute regime (yielding KEEPs at MLP=5 and MLP=6) and gpu6 probed the optimizer team's UNEMBEDDING_LR under the new champion (yielding a third cross-team KEEP). Three of the seven accepted improvements thus came from experiment agents probing research directions another team had closed or skipped — the shared workshop and the post-KEEP cross-team suggestion threads carried the coordination signal that an

analyst would have written. The analyst role appears at-or-below replacement value at this template surface and 50-experiment budget, though with $n = 1$ runs per condition the gap to the full system is not statistically supported and a multi-replicate study ($n \geq 3$) would be required to call it.

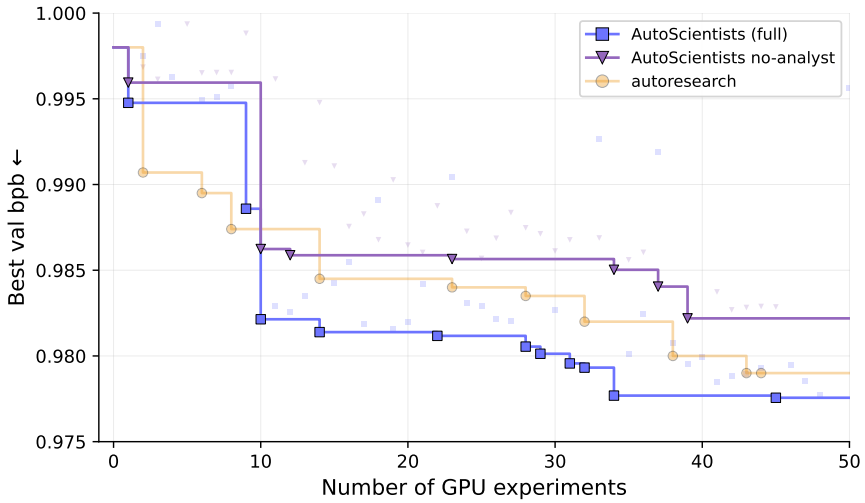


Figure S6: Per-experiment trajectory for the `abl-no-analyst` ablation (purple) compared with the full AUTOSCIENTISTS system (red) and single-agent autoresearch (blue), all anchored at the pristine nanochat baseline (`val_bpb` $= 0.998$). Solid step curves are the running best-so-far validation `val_bpb`; opaque markers indicate accepted KEEPs; transparent markers show per-experiment attempts that did not improve the running min. Final values: autoresearch 0.9773 (15 KEEPs / 83 exps), full AUTOSCIENTISTS 0.9777 (11 KEEPs / 71 exps), `abl-no-analyst` 0.9817 (7 KEEPs / 50 exps).

D.4 Independent Agents (`abl-independent`)

Removing all cross-agent coordination (no comments, no notifications, no suggestions, near-miss reports, discussion posts, gap analyses, or rankings, and no shared artifacts) and reducing the roster to six experiment agents leaves each agent running an independent autoresearch loop with its own private champion seeded from the pristine upstream and no view of any other agent's state. The resulting system reaches a best-of-population `val_bpb` of 0.9833 over 50 unique experiments (Fig. S7), against 11 KEEPs and 0.9777 in 71 experiments for the full system on the same anchor. Whereas `abl-no-cross-agent` forbids talk between agents but still lets them inherit each other's wins through a shared champion program and dead-ends file, this ablation additionally hides those files. The cost of this further isolation is visible in two ways. First, five of the six agents independently rediscovered the same dominant first-axis win (the `TOTAL_BATCH_SIZE` $2^{19} \rightarrow 2^{18}$ reduction also surfaced in `abl-no-cross-agent`) within their first two experiments, spending roughly a third of the budget on duplicates that the shared dead-ends ledger would have suppressed. Second, the agents that escaped the pristine plateau followed disjoint search paths and produced incompatible champions; with no shared surface the population cannot observe these interactions or combine its disjoint wins, and the best-of-population trajectory plateaus roughly 0.006 above the full-system anchor.

E AUTOSCIENTISTS Output on GPT nanochat

On the GPT nanochat training-optimization task, AUTOSCIENTISTS descended from the Autoresearch baseline at `val_bpb` $= 0.998$ to a final champion at `val_bpb` $= 0.97769$ over 75 experiments. Alongside the champion training script, the run produces two structured documents that follow the templates released with the system: a *model card* (Sec. E.1) covering model details, training procedure, evaluation, and compute; and a *research insights* document (Sec. E.2) recording the reasoning trajectory: which research directions were tried, which were accepted, which were rejected, and what the analyst notes recorded as the mechanism in each case. The per-experiment trajectory is the Run 1 trace in Figure S3.

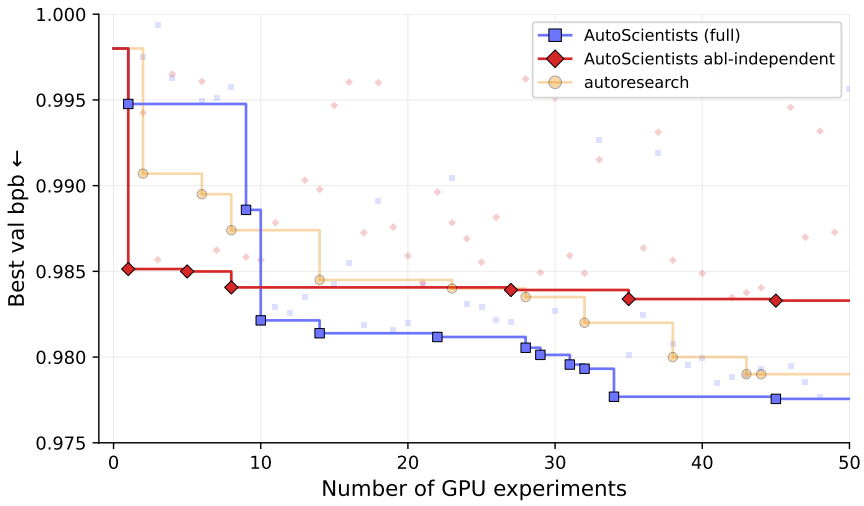


Figure S7: Per-experiment trajectory for the `abl-independent` ablation compared with the full AUTOSCIENTISTS system and single-agent autoresearch, all anchored at the pristine nanochat baseline (`val_bpb` $= 0.998$). Solid step curves are the running best-so-far validation `val_bpb` (population minimum across the six solo loops for `abl-independent`); opaque markers indicate global best-so-far drops; transparent markers show per-experiment attempts that did not improve the running min. Final values: autoresearch 0.9773 (15 KEEPs / 83 exps), full AUTOSCIENTISTS 0.9777 (11 KEEPs / 71 exps), `abl-independent` 0.9833 (best-of-6 over 50 exps).

E.1 Model Card

This card is populated from the champion record. Fields that are not informative at this scale (carbon-emission accounting, demo links, downstream-use guidance) are omitted. An analogous card for a BioML-Bench task is shown in Figure 2 of the main paper.

Model details.

- **Developed by:** AUTOSCIENTISTS multi-agent system (9 agents: 6 experiment, 3 analyst), with Claude Sonnet as the per-agent backend.
- **Model type:** decoder-only transformer language model, 87 M parameters.
- **Language:** English (FineWeb shard).
- **Base model:** pristine nanochat baseline released with Autoresearch [3] (`val_bpb` $= 0.998$).

Uses.

- **Direct use:** a research artifact for short-budget LM-training optimization. The model is below the scale at which broad linguistic competence emerges and is not intended for production language modeling.
- **Out-of-scope use:** downstream language tasks, bias-sensitive applications, and any deployment requiring distributional coverage beyond the FineWeb training shard.

Bias, risks, and limitations. The model is trained on a 40 M-token slice of FineWeb and inherits any biases of that source. The 300 s training budget produces a substantially underfit model relative to publication-scale LMs; `val_bpb` should not be interpreted as a language-task performance score.

Training data. A FineWeb shard distributed with the Autoresearch reference, tokenized by the upstream BPE-8192 tokenizer. The training and evaluation data specifications are inherited verbatim from the task definition shared by all agents.

Training procedure.

- **Architecture knobs:** `DEPTH` $= 7$, `ASPECT_RATIO` $= 96$ (`model_dim` $= 768$), `HEAD_DIM` $=$ 128 (6 heads), `WINDOW_PATTERN` $=$ `SSSL` with short-window ratio $1/8$, RoPE, ResFormer alternating value embeddings.
- **Optimizer:** hybrid Muon (NorMuon variant; Polar-Express normalization at constant 1.0 with 4 Newton–Schulz iterations) for 2D matrix parameters; AdamW for scalars, embeddings, and gates. Per-group learning rates: `EMBEDDING_LR` $= 0.6$, `UNEMBEDDING_LR` $=$ 0.004, `MATRIX_LR` $= 0.04$, `SCALAR_LR` $= 0.5$. `WEIGHT_DECAY` $= 0.2$.
- **Schedule:** `TOTAL_BATCH_SIZE` $= 2^{18}$, `DEVICE_BATCH_SIZE` $= 128$, `WARMUP_RATIO` $= 0$, `WARMDOWN_RATIO` $= 0.5$, `FINAL_LR_FRAC` $= 0$.
- **Training regime:** bf16 mixed precision; `torch.compile` with full-graph capture.

Speeds, sizes, times. 300 s wall-clock training (excluding compilation warmup); $1{,}293$ optimizer steps; 339 M tokens processed; 43.5% model-FLOPs utilization; 58 GB peak VRAM.

Evaluation.

- **Test data:** pinned 40 M-token held-out FineWeb shard, excluded from training.
- **Metric:** validation bits-per-byte, `val_bpb` $=$ total_nats$/(\log 2 \cdot$ total_bytes$)$, with byte-length weighting per token.
- **Result:** `val_bpb` $= 0.97769$, reproducible to within the empirical noise floor ($\sigma = 1.0 \times 10^{-3}$, $n = 7$ cross-seed pairs).

Compute infrastructure. Single NVIDIA H100 80 GB GPU. The training script is self-contained: re-execution requires only the upstream Autoresearch repository plus the script.

E.2 Research Insights

The companion document recording the reasoning trajectory of the run.

Run details.

- **Task:** GPT nanochat training optimization.
- **Starting metric value:** `val_bpb` $= 0.998$.
- **Final metric value:** `val_bpb` $= 0.97769$.
- **Total experiments run:** 75.
- **Experiments accepted (KEEPs):** 8 (1 baseline lock plus 7 improvements; 10.7% acceptance).
- **Wall-clock duration:** ≈ 6.4 hours.
- **Number of agents:** 9 (6 experiment, 3 analyst).
- **Base LLM:** Claude Sonnet (per agent).

Findings. The seven accepted improvements (Table S4) span three distinct mechanism categories rather than concentrating on a single axis: *throughput* (more optimizer steps within the 300 s budget), *capacity* (more parameters per token), and *optimizer quality* (better gradient signal per step). The dominant first-axis win is a throughput effect: halving the per-step batch from 2^{19} to 2^{18} approximately doubles the number of optimizer steps at near-constant total tokens and alone accounts for -0.0090 of the total -0.017 descent. The largest single capacity step bundles a learning-rate correction: widening the model from d=512 to d=768 raises parameter count from 50 M to 94 M and simultaneously brings the d_{model}-LR-scale factor from 0.78 to 1.0. The remaining five wins refine the throughput / quality balance: `ns_steps` $5 \to 4$ in Polar-Express Muon, two halvings of the short-attention-window ratio, removing the $1.02\times$ Polar-Express normalization margin, and a final trade of 8% parameters for 14.4% optimizer steps via depth reduction $8 \to 7$. Each refinement is multi-seed-confirmed against the noise floor.

Table S4: Chain of accepted improvements (KEEPs) produced by AUTOSCIENTISTS on the GPT nanochat task. Each row records the proposing experiment agent, its team, the modified axis and change applied, the resulting validation `val_bpb`, the Δ from the contemporaneous champion, and the mechanism category recorded by the analyst on the proposing team. Row 0 is the run-local baseline lock; rows 1–7 form the productive descent to the champion at `val_bpb` $= 0.977687$. Mechanism categories: *T* = throughput (more optimizer steps within the budget), *C* = capacity (more parameters per token), *Q* = optimizer quality (better gradient signal per step).

#	Agent	Team	Axis	Change	val_bpb	Δ	Cat.	Mechanism (analyst-recorded)
0	Experiment Agent 3	schedule	—	upstream baseline lock	0.99476	—	—	run-local anchor for noise-floor calibration
1	Experiment Agent 1	throughput	`TOTAL_BATCH_SIZE`	$2^{19} \to 2^{18}$	0.98573	-0.00903	T	$\sim 2\times$ optimizer steps at near-constant total tokens
2	Experiment Agent 5	hidden	`ASPECT_RATIO`	$64 \to 96$ ($d_{\text{model}}: 512 \to 768$)	0.98214	-0.00359	C	50M→94M params; d_{model} LR-scale to 1.0 (removes 22% LR overcorrection)
3	Experiment Agent 6	hidden	`ns_steps`	$5 \to 4$	0.98106	-0.00108	Q	4 Newton–Schulz iterations sufficient for Polar-Express orthogonalization
4	Experiment Agent 6	hidden	`short_window_ratio`	$1/2 \to 1/4$	0.98054	-0.00052	T	shorter S-layer window $\to +2.8\%$ optimizer steps
5	Experiment Agent 6	hidden	`short_window_ratio`	$1/4 \to 1/8$	0.98013	-0.00042	T	axis-bracket continuation; $+2.2\%$ steps with second-seed confirmation
6	Experiment Agent 2	throughput	`polar_express_norm`	$1.02 \to 1.0$	0.97956	-0.00056	Q	exact normalization in Polar-Express Muon (removes numerical safety margin)
7	Experiment Agent 4	schedule	`DEPTH`	$8 \to 7$ (at `ASPECT_RATIO` $= 96$)	0.97769	-0.00188	T	-8% params (94M→87M) for $+14.4\%$ optimizer steps (1130→1293)

Generalizable insights.

1. Under a fixed wall-clock training budget, optimizer-step granularity dominates over total-token volume on this task: the same total tokens processed in $\sim 2\times$ as many steps yields a $\sim 1\%$ `val_bpb` reduction. Counterevidence would be a fixed-budget benchmark in which step doubling at constant total tokens produces no improvement.
2. Capacity gains in this regime are gated by accompanying LR-scale corrections; widening the model alone improved `val_bpb` by -0.0036, with the analyst notes attributing most of that to the d_{model}-LR scale becoming exact rather than to raw parameter count. Counterevidence would be a width-up KEEP whose internal LR-scale factor was already exact.
3. Schedule shape is load-bearing: pure shape rewrites (e.g. replacing the linear warmdown with a constant) regress sharply, whereas magnitude changes within the existing functional form are productive.

Task-specific findings.

1. Polar-Express Muon's $1.02\times$ numerical-stability margin is unnecessary at this scale: removing it improves `val_bpb` by -0.000564 at 4 Newton–Schulz iterations. Whether this transfers to other Muon implementations or other model scales is not tested by this run.
2. The `SSSL` window pattern with short-window ratio $1/8$ is dominant over both pure short (`SSSS`) and shifted (`SSLL`) replacements; the analyst notes interpret this as preserving a local / long-range attention balance the alternatives break.

Dead ends and negative results. The run rejects 67 experiments, with one further DISCARD flagged by the multi-seed gate as a near-miss (`short_window_ratio` $= 1/16$, where the two seeds disagreed). Table S5 reports ten representative directions spanning the search space and reveals two qualitatively different kinds of dead end. The first sets sharp limits on the productive region: $|\Delta| > 5\sigma$ regressions on `rotary_base` $=$ 1k, `DEPTH` $= 6$, `HEAD_DIM` $= 64$, `ASPECT_RATIO` $= 112$, and `ns_steps` $= 3$ constrain the champion to a narrow ridge in five orthogonal directions. The second characterizes the ridge rather than bounding it: both `mlp_ratio` directions (3 and 5) are worse than the champion's 4 by similar amounts, all four `WARMDOWN_RATIO` probes $(0.25, 0.30, 0.65, 0.70)$ regress relative to the champion's 0.5, and replacing the `SSSL` window pattern with either `SSSS` or `SSLL` loses the local / long-range balance.

Table S5: Representative dead ends from the AUTOSCIENTISTS run on GPT nanochat. Of 75 logged experiments, 67 were rejected. Ten representative research directions drawn from across the search space are shown here; for each, the row reports the direction tested, the value, the resulting `val_bpb`, Δ from the contemporaneous champion (positive by definition for a DISCARD), and the mechanism the analyst recorded for why the change failed.

Axis	Direction	Value	val_bpb	Δ	Reason
`DEPTH`	decrease	6 (`ASPECT_RATIO` $= 64$)	1.0226	$+0.0278$	capacity collapse: 50M→26M params (-48%) outweighs throughput
`HEAD_DIM`	decrease	64	0.9904	$+0.0094$	lower-quality attention scores dominate over throughput gain
`ASPECT_RATIO`	increase	112	0.9877	$+0.0100$	per-step compute outpaces capacity gain at fixed budget
`TOTAL_BATCH_SIZE`	decrease	2^{17}	0.9913	$+0.0056$	gradient-variance penalty exceeds step-count gain
`WINDOW_PATTERN`	replace	SSSS, SSLL	0.9808, 0.9829	$+0.0031$, $+0.0019$	SSSL is load-bearing; pure short / shifted patterns lose long-range signal
`rotary_base`	both	1k, 100k	0.9876, 0.9826	$+0.0055$, $+0.0004$	frequency-band mismatch with 2k-token training context
`ns_steps`	decrease	3	0.9891	$+0.0080$	insufficient orthogonalization quality (opposite direction from KEEP at 4)
`WARMDOWN_RATIO`	both	0.25, 0.30, 0.65, 0.70	0.9856–0.9979	$+0.0010$ to $+0.0099$	champion 0.5 sits at a genuine optimum; both shorter and longer warmdown waste gradient signal
`softcap`	decrease	30, 10	0.9994, 0.9951	$+0.0046$, $+0.0003$	logit softcap clips informative high-confidence logits
`mlp_ratio`	both	3, 5	0.9801, 0.9801	$+0.0006$, $+0.0024$	narrow local optimum at champion's value of 4

Coordination and team dynamics. The cold-start discussion produced three teams (architecture, schedule, throughput) over 12 substantive forum posts and a $6/6$ DONE / MORE vote split, with no predefined axis assignment. The accepted improvements distribute across all three teams (architecture: 4, throughput: 2, schedule: 2). Cross-team transfer is observable in the chain: the short-attention-window axis was bracketed by the architecture team, the Polar-Express normalization refinement was proposed by the throughput team on the resulting wider-model champion, and the final depth reduction was proposed by the schedule team on top of the throughput team's polar-norm KEEP. The multi-seed gate prevented one near-miss promotion (`short_window_ratio` $= 1/16$), where the two seeds disagreed.

Limitations of these insights. The findings come from a single run of the system on a single task with a single agent backend at the 300 s budget. The throughput-dominance claim is most confidently established by the -0.009 first-axis effect; the smaller refinement effects (≤ 0.002) are individually multi-seed-confirmed but their compounding behavior under different orderings is untested, since each refinement was applied to the contemporaneous champion rather than to the pristine baseline. Several research directions were not probed, including pretraining-objective changes, weight initialization beyond the embedding-init scales tested, and data-mixture ratios.

F Implementation Details of BioML-Bench

F.1 Setup

Experiment compute resources. In the BioML-Bench protocol drug discovery, protein engineering, and single cell omics tasks are run on CPU-only machines with an 8-hour limit. In contrast, we run AUTOSCIENTISTS, Autoresearch, and rerun Biomni [10] under a unified experimental-compute setting of 4 hours on 1 H100 GPU with 16 CPUs and 48 GB memory for all tasks except for biomedical imaging which had a wall-clock budget of 16 hours instead. We therefore compare against the published BioML-Bench baselines while noting that results for other baselines on drug discovery, protein engineering, and single cell omics tasks are obtained under a slightly modified hardware protocol. We adopt this unified setting to better reflect contemporary biomedical ML practice, where GPU-backed environments are commonly available even for non-imaging workloads while imposing a shorter wall-clock budget. Additionally, the agents did not use web search or fetch in any of their experiments.

Leaderboarding. We adapt the BioML-Bench task descriptions so that the benchmark can be run as an iterative development loop on validation set performance, reflecting how biology-ML research would be approached by a human scientist. The agent must write its own model training script, evaluate it on the validation set, report validation performance to a leaderboard, and submit a final submission on the test set. For each task, validation sets are defined using domain-specific splitting strategies from the BioML-Bench training set: scaffold-based holdout splits for drug discovery, patient-level held-out folds for biomedical imaging, and batch- or site-stratified holdouts for single-cell omics. Final benchmark performance is reported on a held-out BioML-Bench test. We use this approach for AUTOSCIENTISTS as ℓ_{eval}, and for Biomni and Autoresearch.

F.2 Task Datasets and Evaluation Metrics

F.2.1 Biomedical Imaging

kaggle-histopathologic-cancer-detection. PatchCamelyon (PCam): 96×96 RGB pathology image patches with a binary metastatic-cancer label defined by the centre 32×32 region. Training uses $174{,}464$ labelled patches and the test set is the $45{,}561$ patches. The validation set is a random split of the training set. The metric is ROC-AUC (higher is better).

kaggle-osic-pulmonary-fibrosis-progression. A clinical time-series task: 158 idiopathic-pulmonary-fibrosis patients in the training set (each with multi-week FVC measurements and a baseline CT scan). There are 18 test patients with baseline visit only and CT scans. The test grader scores $1{,}908$ `Patient_Week` predictions covering weeks -12 to 133 for the 18 test patients. A patient-level cross-validation split is used for model development. The metric is the modified Laplace log-likelihood $-\sqrt{2}\,\Delta/\sigma_c - \log(\sqrt{2}\,\sigma_c)$ with $\Delta = \min(|FVC_{\text{true}} - FVC_{\text{pred}}|, 1000)$ and $\sigma_c = \max(\sigma, 70)$, averaged over all test (patient, week) pairs (higher is better).

kaggle-rsna-miccai-brain-tumor-radiogenomic-classification. Multi-parametric brain MRI for the RSNA-MICCAI BraTS challenge: each patient has four DICOM sequences (FLAIR, T1w, T1wCE, T2w). Training uses the 526 patients with binary MGMT promoter methylation labels and the test set is 59 patients. Validation uses a patient-level cross-validation split. The metric is ROC-AUC of the predicted MGMT-methylation probability (higher is better).

kaggle-uw-madison-gi-tract-image-segmentation. MR-Linac guided MRI for radiation-therapy planning: 50 cancer patients imaged across 1–5 treatment days, with stomach / small-bowel / large-bowel run-length-encoded segmentation masks. Provided are 77,328 slice-class rows in the training set and a 17,760-row validation set with held-out patients and held-out treatment dates. The test set contains 20,400 entries (6,800 slices $\times$ 3 classes). The metric is $0.4\,\mathrm{Dice} + 0.6\,(1 - \mathrm{HD}_{3D})$, where Dice is per (slice, class) (with 0 for the empty-pred / empty-truth case) and HD_{3D} is the symmetric Hausdorff distance of the class-OR-merged 3D case-day volume, normalised by the unit-cube diagonal $\sqrt{3}$ to lie in $[0, 1]$ (higher overall score is better).

F.2.2 Drug Discovery

All drug-discovery tasks share the same supervised structure: a public training set and a held-out, label-blinded test set, both downloaded from the Polaris Hub. During development the training set is split into five Murcko-scaffold groups and 5-fold scaffold cross-validation is used as ℓ_{eval}.

tdcommons-bbb-martins. Binary classification of blood–brain-barrier (BBB) permeability for 1,624 training molecules and 406 test molecules. The metric is ROC-AUC (higher is better).

tdcommons-caco2-wang. Regression of Caco-2 cell permeability ($\log P_{\text{app}}$) for 728 training molecules and 182 test molecules. The metric is mean absolute error (MAE; lower is better).

tdcommons-cyp2d6-substrate-carbonmangels. Binary classification of CYP2D6 substrate activity for 532 training molecules and 135 test molecules. The metric is PR-AUC (higher is better) to remain meaningful under imbalance.

tdcommons-herg. Binary classification of hERG potassium-channel inhibition for 523 training molecules and 132 test molecules. The metric is ROC-AUC (higher is better).

tdcommons-lipophilicity-astrazeneca. Regression of octanol/water lipophilicity ($\log D_{7.4}$) for 3,360 training molecules and 840 test molecules from the AstraZeneca DMPK release. The metric is MAE (lower is better).

polaris-adme-fang-hclint-1. Regression of human-liver-microsome intrinsic clearance (`LOG_HLM_CLint`, mL/min/kg) on the `adme-fang-hclint-1` benchmark with 2,229 training molecules and 575 test molecules. The metric is Pearson correlation r between predicted and measured log clearance (higher is better).

polaris-adme-fang-hppb-1. Regression of human plasma-protein binding (`LOG_HPPB`, % unbound) on the `adme-fang-hppb-1`. There are 126 training molecules and 34 test molecules. The metric is Pearson r (higher is better).

polaris-adme-fang-solu-1. Regression of aqueous solubility (`LOG_SOLUBILITY`) on the `adme-fang-solu-1` benchmark with 1,578 training molecules and 400 test molecules. The metric is Pearson r (higher is better).

polaris-pkis2-egfr-wt-c-1. Binary classification of EGFR wild-type kinase inhibition (`CLASS_EGFR`) on the `pkis2-egfr-wt-c-1` benchmark with 496 training compounds and 144 test compounds and severe class imbalance. The metric is PR-AUC (higher is better) to remain meaningful under imbalance.

F.2.3 Single Cell Omics

open-problems-predict-modality. A bone-marrow mononuclear cell (BMMC) CITE-seq dataset in which paired RNA expression ($\sim$13,000 genes) and surface-protein abundance ($\sim$134 proteins) are measured in the same cells. Validation are held-out site/donor batches. The metric is RMSE between predicted and measured protein values across all (cell, protein) pairs (lower is better).

open-problems-single-cell-perturbations. A PBMC perturbation screen with 144 compounds and 5,317 genes, in which differential-expression profiles (`clipped_sign_log10_pval` clipped to $[-4, 4]$) are measured per (compound, cell-type) pair. Training uses the T-cell, NK-cell and regulatory-T-cell rows. Validation is a 20% within-cell-type split of those training rows. Testing predicts the 151 (compound, cell-type) profiles for B cells and Myeloid cells. The metric is Mean Rowwise RMSE (MRRMSE; lower is better).

open-problems-cell-cell-communication-ligand-target. This task contains a triple-negative-breast-cancer single-cell RNA-seq dataset accompanied by an OmniPath ligand–receptor prior. The supervised labels are extremely sparse: 81 labelled (ligand, target-cell-type) pairs and 731 unlabelled test pairs. The validation metric is a random $80/20$ split of the 81 labelled pairs. The metric is the odds ratio of true positives in the top-5% scored pairs against the held-out binary response, with the 0.5-shrinkage formula score $= 1 - 1/(1 + \mathrm{OR}/2)$ for boundedness (higher is better).

open-problems-label-projection. A diabetic kidney-disease single-nucleus RNA-seq dataset with 13 cell types. The validation set is a held-out batch. The metric is the weighted F1-score across cell types (higher is better).

open-problems-spatially-variable-genes. A SlideSeqV2 mouse cortex spatial-transcriptomics dataset covering 210 genes and a smaller cerebellum dataset with labels is provided. The task is unsupervised and cerebellum labels may be used to verify that an unsupervised statistic correlates with ground truth. The metric is Kendall's τ between predicted spatial scores and the held-out continuous `spatial_var_score` for the 210 cortex genes (higher is better).

F.2.4 Protein Engineering

ProteinGym datasets do not provide a held-out test set: the out-of-fold predictions from a prescribed 5-fold cross-validation are themselves the official evaluation, with three split strategies (`fold_random_5`, `fold_modulo_5`, `fold_contiguous_5`) for substitution scans and only `fold_random_5` for indel scans. The final score is the mean Spearman correlation between predicted and measured fitness, computed in raw target space (higher is better). The four substitution-mutation DMS datasets are: SPIKE_SARS2_Starr_2020_binding, SBI_STAAM_Tsuboyama_2023_2JVG, CBX4_HUMAN_Tsuboyama_2023_2K28, PSAE_PICP2_Tsuboyama_2023_1PSE, and the two indel datasets are: Q8EG35_SHEON_Campbell_2022 and CSN4_MOUSE_Tsuboyama_2023_1UFM.

Table S6: Performance on four domains in BioML-Bench. We report the published performance of Reference, MLAgentBench, AIDE, and STELLA from [2]. Detailed results are available in Table S7. Values are mean (SE) across tasks.

Domain	Agent	Leaderboard Percentile (↑)	Mean Rank (↓) ¶	Above Median (% ↑)	Any Medal (% ↑)	Completion Rate (% ↑)
Biomedical Imaging ($n = 4$)	Reference	3.30 (2.52)	–	0.0 (0.0)	0.0 (0.0)	NA
	MLAgentBench°	21.73 (11.41)	–	12.5 (12.5)	0.0 (0.0)	100.0
	AIDE°	12.11 (9.43)	–	6.2 (6.2)	0.0 (0.0)	81.2
	STELLA°	5.91 (4.98)	–	6.2 (6.2)	0.0 (0.0)	68.8
	Biomni°	19.04 (10.83)	3.00	12.5 (12.5)	12.5 (12.5)	100.0
	Autoresearch°	39.60 (21.75)	1.75	25.0 (25.0)	25.0 (25.0)	100.0
	AUTOSCIENTISTS °	**45.75 (22.18)**	**1.25**	**50.0 (28.9)**	**50.0 (28.9)**	100.0
Drug Discovery ($n = 9$)	Reference	1.11 (1.11)	–	0.0 (0.0)	0.0 (0.0)	NA
	MLAgentBench*	22.45 (5.93)	–	16.7 (9.3)	5.6 (3.7)	100.0
	AIDE*	24.75 (7.23)	–	25.0 (11.0)	5.6 (5.6)	80.6
	STELLA*	28.84 (7.84)	–	25.0 (11.0)	13.9 (7.3)	100.0
	Biomni†	47.91 (10.77)	2.22	44.4 (17.6)	44.4 (17.6)	100.0
	Autoresearch†	46.16 (10.59)	2.00	33.3 (16.7)	33.3 (16.7)	100.0
	AUTOSCIENTISTS †	**64.52 (8.37)**	**1.78**	**55.6 (17.6)**	**55.6 (17.6)**	100.0
Protein Engineering ($n = 6$)	Reference	0.00 (0.00)	–	0.0 (0.0)	0.0 (0.0)	NA
	MLAgentBench*	13.52 (9.18)	–	12.5 (12.5)	0.0 (0.0)	100.0
	AIDE*	24.50 (6.90)	–	25.0 (9.1)	12.5 (8.5)	75.0
	STELLA*	34.98 (12.51)	–	45.8 (18.7)	16.7 (12.4)	100.0
	Biomni†	93.94 (3.83)	2.50	**100.0 (0.0)**	**100.0 (0.0)**	100.0
	Autoresearch†	**96.97 (3.03)**	2.00	**100.0 (0.0)**	**100.0 (0.0)**	100.0
	AUTOSCIENTISTS †	**96.97 (3.03)**	**1.50**	**100.0 (0.0)**	**100.0 (0.0)**	100.0
Single Cell Omics ($n = 5$)	Reference	7.34 (4.52)	–	0.0 (0.0)	0.0 (0.0)	NA
	MLAgentBench*	28.83 (13.04)	–	25.0 (13.7)	20.0 (14.6)	90.0
	AIDE*	53.17 (14.86)	–	55.0 (16.6)	35.0 (18.7)	85.0
	STELLA*	54.23 (14.09)	–	60.0 (15.0)	40.0 (20.3)	85.0
	Biomni†	78.00 (10.20)	2.60	80.0 (20.0)	**80.0 (20.0)**	100.0
	Autoresearch †	86.00 (9.80)	1.80	**100.0 (0.0)**	**80.0 (20.0)**	100.0
	AUTOSCIENTISTS †	**88.00 (9.70)**	**1.60**	**100.0 (0.0)**	**80.0 (20.0)**	100.0

¶ Mean rank is computed only among Biomni, Autoresearch, and AUTOSCIENTISTS since their experimental-compute budgets are matched. These methods also represent the strongest-performing approaches overall.
$*\circ\dagger$ Wall-clock time and compute access of agents: $*$ for 8h CPU, † for 4h GPU & CPU, and ○ for 16h GPU & CPU.

Table S7: Per-task comparison of Biomni, Autoresearch, and AUTOSCIENTISTS on BioML-Bench. For each method we report the task score, leaderboard percentile (LB%), whether the score is above the median leaderboard entry (Med), and medal status. The best score is shown in **bold**.

		Biomni				Autoresearch				AUTOSCIENTISTS			
Task	**Metric**	**Score**	**LB%**	**Med**	**Medal**	**Score**	**LB%**	**Med**	**Medal**	**Score**	**LB%**	**Med**	**Medal**
Biomedical imaging													
kaggle-histopathologic-cancer-detection	ROC-AUC	0.81818	20.8	N	–	0.99832	99.3	Y	Gold	**0.99834**	99.3	Y	Gold
kaggle-osic-pulmonary-fibrosis-progression	Laplace LL↑	-9.42451	5.0	N	–	-7.51872	9.8	N	–	**-7.11904**	13.9	N	–
kaggle-rsna-miccai-brain-tumor-radiogenomic-classification	ROC-AUC	0.51338	49.0	N	–	0.52353	44.3	N	–	**0.54353**	64.8	Y	Bronze
kaggle-uw-madison-gi-tract-image-segmentation	Dice+HD↑	0.20356	1.4	N	–	**0.56327**	5.0	N	–	0.55114	5.0	N	–
Drug discovery													
tdcommons-bbb-martins	AUROC	0.91014	37.5	N	–	0.90908	25.0	N	–	**0.92030**	75.0	Y	Bronze
tdcommons-caco2-wang	MAE↓	**0.27560**	100.0	Y	Gold	0.32257	25.0	N	–	0.27663	100.0	Y	Gold
tdcommons-cyp2d6-substrate-carbonmangels	AUPRC	0.63100	22.2	N	–	**0.67413**	22.2	N	–	0.61870	22.2	N	–
tdcommons-herg	AUROC	0.85464	71.4	Y	Silver	0.79985	14.3	N	–	**0.86672**	85.7	Y	Silver
tdcommons-lipophilicity-astrazeneca	MAE↓	0.54186	0.0	N	–	**0.40003**	88.9	Y	Gold	0.42191	77.8	Y	Bronze
polaris-adme-fang-hclint-1	Pearson r	0.71530	60.0	Y	Bronze	**0.73308**	90.0	Y	Gold	0.69122	50.0	N	–
polaris-adme-fang-hppb-1	Pearson r	0.84818	80.0	Y	Silver	0.83560	80.0	Y	Bronze	**0.87292**	80.0	Y	Silver
polaris-adme-fang-solu-1	Pearson r	0.64745	20.0	N	–	0.65073	20.0	N	–	**0.65778**	50.0	N	–
polaris-pkis2-egfr-wt-c-1	PR-AUC	0.74681	40.0	N	–	**0.77942**	50.0	N	–	0.76616	40.0	N	–
Single Cell Omics													
open-problems-predict-modality	RMSE↓	**0.59122**	100.0	Y	Gold	0.64356	100.0	Y	Gold	0.69722	100.0	Y	Gold
open-problems-single-cell-perturbations	MRRMSE↓	0.78246	80.0	Y	Silver	0.78163	80.0	Y	Silver	**0.77241**	90.0	Y	Gold
open-problems-cell-cell-communication-ligand-target	Odds ratio↑	0.68281	100.0	Y	Gold	0.70835	100.0	Y	Gold	**0.92367**	100.0	Y	Gold
open-problems-label-projection	F1-weighted	0.95437	60.0	Y	Bronze	**0.97484**	100.0	Y	Gold	0.96394	100.0	Y	Gold
open-problems-spatially-variable-genes	Kendall τ	0.64992	50.0	N	–	0.68923	50.0	Y	–	**0.69631**	50.0	Y	–
Protein engineering													
proteingym-dms-SPIKE_SARS2_Starr_2020_binding	Spearman↑	0.59660	81.8	Y	Silver	0.59294	81.8	Y	Silver	**0.67011**	81.8	Y	Silver
proteingym-dms-SBI_STAAM_Tsuboyama_2023_2JVG	Spearman↑	0.80046	81.8	Y	Silver	0.81837	100.0	Y	Gold	**0.83384**	100.0	Y	Gold
proteingym-dms-CBX4_HUMAN_Tsuboyama_2023_2K28	Spearman↑	0.94496	100.0	Y	Gold	0.95137	100.0	Y	Gold	**0.95676**	100.0	Y	Gold
proteingym-dms-PSAE_PICP2_Tsuboyama_2023_1PSE	Spearman↑	0.93362	100.0	Y	Gold	0.90897	100.0	Y	Gold	**0.97606**	100.0	Y	Gold
proteingym-dms-Q8EG35_SHEON_Campbell_2022_indels	Spearman↑	0.80133	100.0	Y	Gold	**0.82683**	100.0	Y	Gold	0.81119	100.0	Y	Gold
proteingym-dms-CSN4_MOUSE_Tsuboyama_2023_1UFM_indels	Spearman↑	0.93493	100.0	Y	Gold	**0.93965**	100.0	Y	Gold	0.93186	100.0	Y	Gold

F.3 Performance Across Independent Runs

Due to the computationally intensive nature of running BioML-Bench, it was not feasible to repeat all experiments with multiple random initializations. Instead, we assess variability across independent runs on a representative task. We selected tdcommons-herg, which has a time-budget of 4 hours, and ran 3 independent runs of AUTOSCIENTISTS. Performance of the three runs achieved an AUROC of 0.867, 0.830, and 0.862, respectively. We observe that performance is relatively stable across independent runs, with a mean of 0.853 and standard deviation of 0.020. Additionally, for all three runs AUTOSCIENTISTS ranks first compared to Biomni and Autoresearch (Table S7).

F.4 Performance Across Wall-Clock Time

Here we plot mean Spearman ρ over wall-clock time for AUTOSCIENTISTS v.s. Autoresearch on six BioML-Bench Protein Engineering tasks. We draw this comparison for the Protein Engineering subset of BioML-Bench because each task reports the same mean Spearman ρ averaged across the same set of CV splits for ℓ_{eval} and for the final leaderboard performance. So tracking ℓ_{eval}, the mean Spearman ρ over wall-clock time, captures progress on the final leaderboard. Conversely, the other Bio-ML Bench tasks instead score a held-out test submission, which evaluates how well the agent searched the model space and how well its final approach generalizes to unseen data. For the Protein Engineering tasks, we show for the given 4h budget, how quickly the champion of each approach improves (Fig. S8).

F.5 Behavior of AUTOSCIENTISTS on BioML-Bench Tasks

Here we analyse the approaches taken by AUTOSCIENTISTS. Each task's pipeline is classified into seven non-exclusive method categories: **(1) Gradient-boosted trees** (XGBoost, LightGBM, CatBoost, ExtraTrees, RandomForest); **(2) Foundation-model frozen features** (using a pretrained protein/molecule/image model only as a fixed feature extractor); **(3) Foundation-model fine-tuning** (LoRA on ESM-2, full ImageNet-CNN fine-tuning, end-to-end ChemBERTa); **(4) Custom neural net trained from scratch** (MLP, ResMLP, Chemprop D-MPNN, U-Net without pretraining); **(5) Kernel and instance methods** (RBF-SVM, Tanimoto-SVM, Gaussian-process with Tanimoto kernel, SVR, k-NN); **(6) Linear/regularised-linear models** (Ridge, RidgeCV, Lasso, LogisticRegression — usually as a meta-learner); **(7) Hand-crafted heuristics** (no learned model).

Biomedical Imaging (4 tasks). Three of four imaging tasks centre on ImageNet-pretrained CNNs fine-tuned end-to-end: `histopathologic-cancer` fine-tunes a dual-stream EfficientNet-B3 [69] with a 3072→512→128→1 fusion head, MixUp, and 4-rotation TTA; `rsna-brain-tumor` fine-tunes EfficientNet-B0 [69] augmented with a slice attention pool and a two-layer Transformer en-

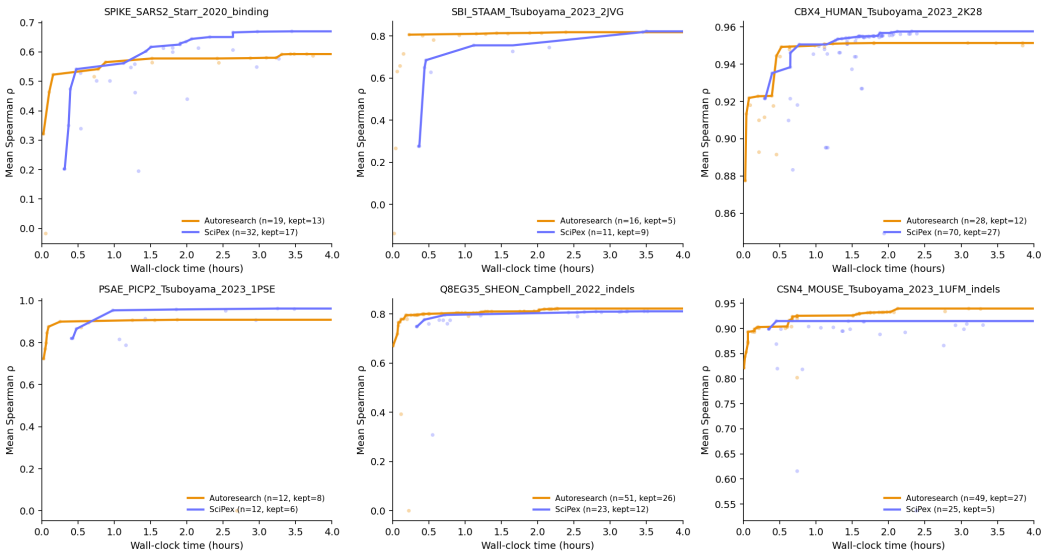


Figure S8: Champion mean Spearman ρ over wall-clock time for AUTOSCIENTISTS v.s. Autoresearch on six BioML-Bench Protein Engineering tasks. The solid line tracks the current champion, and the points show experiments that were run but did not outperform the current champion.

coder; `uw-madison-gi` fine-tunes a `segmentation_models_pytorch` U-Net with an EfficientNet-B4 [69] encoder for multi-organ segmentation. Foundation-model fine-tuning therefore accounts for 3/4 of imaging tasks. The fourth task, `osic-pulmonary-fibrosis`, is architecturally distinct: it uses pre-extracted CNN deep features as a fixed embedding (FM-frozen), fits Ridge regressions for FVC slope and curvature (Linear/Ridge), and retrieves nearest neighbours in PCA-50 CT feature space (Kernel/k-NN). No imaging task uses gradient boosting, a custom NN from scratch, or a heuristic pipeline.

Drug Discovery (9 tasks). Drug-discovery pipelines are dominated by gradient-boosted trees on hand-crafted RDKit fingerprint stacks (6/9). The three tasks that omit boosting entirely are `lipophilicity` (ChemBERTa-2 fine-tuned end-to-end), `hclint` (a 10-seed residual MLP ensemble on Morgan + MACCS + AtomPair + Mordred features), and `solu` (Chemprop v2 D-MPNN, 5-fold scaffold CV). Chemistry-specific D-MPNNs (Chemprop) appear in three tasks: `solu` and `pkis2-egfr` use them as primary or co-primary models, and `hclint` uses a residual MLP on fingerprint features and together with `hclint`'s ResMLP these account for the 3/9 NN-scratch tasks. Only `hppb` uses a foundation model in frozen-feature mode. The AUTOSCIENTISTS approach for `hppb` appends ChemBERTa-2 embeddings (PCA-reduced to 32 dimensions) to a 10-model fingerprint stack before a Ridge meta-learner. `lipophilicity` is the sole task that fine-tunes a foundation model (ChemBERTa-2 via `RobertaForMaskedLM`, CLS-token head). Kernel methods appear in three tasks: `cyp2d6` adds a calibrated RBF-SVM to its CatBoost+LGB+XGB+ET stack, `hppb` includes three SVR variants (RBF $C \in \{1, 5\}$ and linear), and `pkis2-egfr` uses a Tanimoto-SVM and a Tanimoto-GP alongside four boosting models and a Chemprop MPNN. Linear/Ridge models appear as stack meta-learners in four tasks (`hERG`, `cyp2d6`, `BBB`, `hppb`).

Single Cell Omics (5 tasks). Two of five tasks are purely hand-crafted, training-free pipelines: `spatially-variable-genes` builds a 24-signal spatial statistics ensemble (Moran's I, Geary's C, SPARK-X, Getis-Ord G^*, variogram, and others) with weight optimization via differential evolution plus coordinate descent; `cell-cell-communication` scores ligand-receptor pairs using proportion and specificity heuristics with a fine grid search over weights. Two tasks use linear models as their core predictor: `single-cell-perturbations` fits per-gene Ridge regressors on four interpretable features (T cell, NK cell, Treg expression levels, and Tanimoto drug similarity), tuned via leave-one-out CV; `label-projection` applies a multinomial LogisticRegression probe to PCA-reduced HVG features after batch correction. The remaining task, `predict-modality`, trains a residual MLP (TruncatedSVD input $\rightarrow$ 1024 $\rightarrow$ 512 $\rightarrow$ 256 with skip connections, BatchNorm, and Dropout)

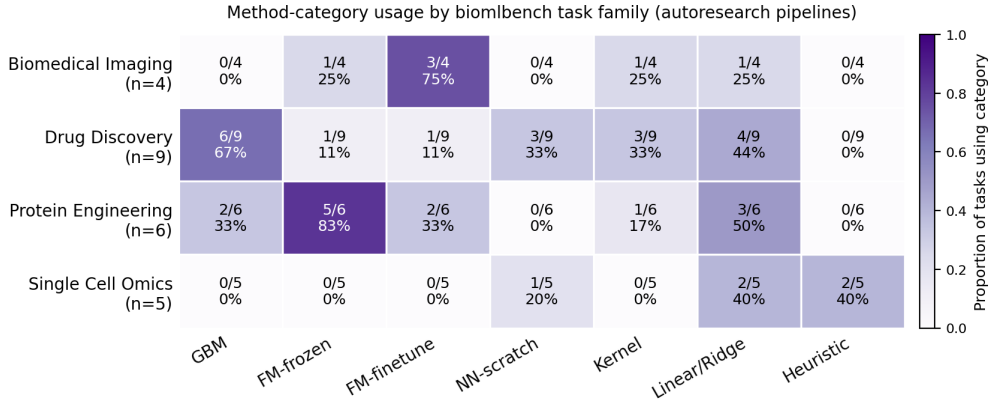


Figure S9: Proportion of tasks within each BioML-Bench task type that uses each method category. The cell value is the fraction of tasks in that row's family in which at least one component of the AUTOSCIENTISTS approach falls in that column's category and categories are non-exclusive (most tasks use several). GBM = gradient-boosted trees; FM-frozen / FM-finetune = pretrained foundation model used as frozen features / fine-tuned; NN-scratch = neural net trained from scratch; Kernel = kernel ridge; Linear/Ridge = Ridge, RidgeCV, Lasso; Heuristic = no learned model.

from scratch for RNA-to-protein prediction. No single-cell task uses gradient boosting, a pretrained foundation model, or a kernel method.

Protein Engineering (6 tasks). Five of six ProteinGym pipelines rely on frozen ESM-2 embeddings (`SPIKE_SARS2`, `SBI_STAAM`, `Q8EG35`, `CBX4_HUMAN`, `CSN4_MOUSE`); only `PSAE_PICP2` is built from fine-tuned models exclusively. Two tasks use ESM-2 LoRA fine-tuning as a component: `CSN4_MOUSE` blends a LoRA model alongside frozen-embedding variants, while `PSAE_PICP2` is a pure seven-way LoRA ensemble (ESM-2-35M and ESM-2-8M). The dominant downstream regressor is Ridge or stacked ridge variants (3/6: `SPIKE_SARS2`, `SBI_STAAM`, `Q8EG35`), followed by gradient boosting as an ensemble member (2/6: `SPIKE_SARS2` includes a LightGBM sub-model; `Q8EG35` includes XGBoost and LGB sub-models). Kernel methods appear only in `Q8EG35`, whose 15-model ensemble includes three SVR variants on ESM-2 features. No ProteinGym pipeline trains a neural net from scratch. Every MLP in the ensembles sits on top of pre-computed ESM-2 embeddings, placing them in the FM-frozen category.

Overall trends. Across 24 tasks, Linear/Ridge models are the most frequently occurring category (10/24), but almost always as meta-learners or simple downstream probes on top of richer representations rather than standalone predictors. Gradient-boosted trees on hand-crafted descriptors are the most common *primary* modelling strategy (8/24), concentrated in drug discovery (6/9). Pretrained foundation models are used either as frozen feature extractors (7/24, predominantly in protein engineering) or fine-tuned end-to-end (6/24, predominantly in biomedical imaging and one small-molecule task). Kernel methods and custom NNs from scratch each appear in roughly a fifth of tasks (5/24 and 4/24, respectively) and nearly always as components within ensembles. Hand-crafted, training-free pipelines win on the two Single Cell Omics tasks where supervised learning is infeasible due to absence of meaningful labelled training signal. Classical molecular fingerprints (Morgan/MACCS/AtomPair/Torsion/RDKit descriptors) are present in every chemistry pipeline that does not fine-tune a foundation model. Fig. S9 shows the proportion of tasks within each BioML-Bench task family that uses each of the seven method categories.

G AUTOSCIENTISTS-Kermut for ProteinGym ACE2-Spike binding DMS

Our AUTOSCIENTISTS-Kermut champion ProteinGym predictor extends the Kermut Gaussian-process model [53] into a three-GP ensemble whose components share a Kermut-style structure kernel but differ in their sequence-side feature sets and kernel families. All hyperparameters (kernel forms, priors, ensemble composition, region-aware noise schedule, and 1000-step optimisation recipe) were chosen on the SARS2-Spike binding task during system development and then frozen. The same configuration is applied unchanged to every other ProteinGym single-substitution protein.

Table S8: Per-task LLM token usage and estimated cost ($ USD) on BioML-Bench. Token counts are reported in millions of tokens (MTok). Costs are estimated using Claude Sonnet 4.6 May 2026 pricing with cache writes charged at the 1-hour cache-write price, and exclude GPU compute.

	Autoresearch					AutoScientists				
Task	Input	Output	CacheCreate	CacheRead	Cost ($)	Input	Output	CacheCreate	CacheRead	Cost ($)
kaggle-histopathologic-cancer-detection	0.001	0.4	0.8	100.0	40.1	0.005	0.8	19.6	232.4	199.6
kaggle-osic-pulmonary-fibrosis-progression	0.006	4.8	5.9	535.7	267.9	0.018	5.7	32.8	738.3	503.9
kaggle-rsna-miccai-brain-tumor-radiogenomic-classification	0.002	0.8	1.7	103.4	52.9	0.010	2.9	27.9	491.4	358.0
kaggle-uw-madison-gi-tract-image-segmentation	0.002	1.1	1.7	146.8	70.3	0.008	0.9	10.7	156.1	124.5
open-problems-cell-cell-communication-ligand-target	0.000	0.4	0.6	26.2	17.6	0.005	1.6	8.5	191.3	133.1
open-problems-label-projection	0.000	0.4	0.4	30.9	16.9	0.002	0.6	4.7	79.2	60.5
open-problems-predict-modality	0.000	0.1	0.1	10.2	5.5	0.003	0.7	5.9	165.5	95.0
open-problems-single-cell-perturbations	0.000	0.3	0.4	23.1	13.9	0.007	2.1	13.2	320.0	206.3
open-problems-spatially-variable-genes	0.003	1.8	3.5	219.8	114.0	0.008	2.6	13.4	317.2	214.2
polaris-adme-fang-hclint-1	0.000	0.1	0.2	18.2	8.4	0.007	0.8	7.7	165.6	108.0
polaris-adme-fang-hppb-1	0.000	0.2	0.3	36.7	16.4	0.007	1.7	11.4	419.5	220.4
polaris-adme-fang-solu-1	0.000	0.2	0.2	28.2	12.3	0.003	0.7	5.4	96.9	71.7
polaris-pkis2-egfr-wt-c-1	0.001	0.8	1.1	130.9	58.0	0.007	1.9	14.3	331.9	213.2
proteingym-dms-CBX4_HUMAN_Tsuboyama_2023_2K28	0.001	0.4	0.5	44.2	22.8	0.006	1.5	9.5	286.1	166.2
proteingym-dms-CSN4_MOUSE_Tsuboyama_2023_1UFM_indels	0.001	0.3	0.4	34.4	17.2	0.003	0.7	5.7	106.4	77.5
proteingym-dms-PSAE_PICP2_Tsuboyama_2023_1PSE	0.000	0.1	0.2	9.7	6.0	0.006	1.4	12.1	286.1	179.4
proteingym-dms-Q8EG35_SHEON_Campbell_2022_indels	0.001	1.6	1.7	121.8	70.0	0.005	1.3	10.2	206.2	143.1
proteingym-dms-SBI_STAAM_Tsuboyama_2023_2JVG	0.000	0.3	0.3	19.4	13.0	0.003	0.8	6.7	149.3	96.4
proteingym-dms-SPIKE_SARS2_Starr_2020_binding	0.000	0.6	0.6	25.8	20.6	0.005	1.4	8.9	174.0	126.1
tdcommons-bbb-martins	0.000	0.1	0.2	21.1	9.3	0.009	1.6	12.2	356.2	203.7
tdcommons-caco2-wang	0.000	0.2	0.2	17.7	8.9	0.006	1.3	7.6	398.8	184.2
tdcommons-cyp2d6-substrate-carbonmangels	0.000	0.1	0.2	22.4	9.7	0.014	1.5	9.3	303.1	168.5
tdcommons-herg	0.000	0.2	0.3	33.2	15.2	0.002	0.7	4.5	69.1	57.4
tdcommons-lipophilicity-astrazeneca	0.001	0.4	0.6	47.4	23.2	0.003	0.5	3.9	143.7	73.9
Total	0.024	15.7	21.9	1807.5	910.1	0.153	35.5	266.1	6184.5	3984.7

G.1 Setup

Inputs and per-fold preprocessing. For each variant, AutoScientists-Kermut reads (i) a 1280-dimensional ESM-2 `t33_650M_UR50D` mean-pool embedding, (ii) the wild-type-marginal masked log-probabilities at the mutated position from a ProteinMPNN conditional-probability map, used both to form $(\log p_{\text{mut}}, \log p_{\text{wt}}, \log p_{\text{mut}} - \log p_{\text{wt}})$ scalars and as the basis of the structure kernel, (iii) the ESM-2 zero-shot fitness score, and (iv) fifteen additional zero-shot predictors covering inverse-folding, structure-aware, MSA-based and homology-based methods (MIF [71], VenusREM [72], ProSST-128/2048/4096 [73], RSALOR [74], ESCOTT [75], xTrimoPGLM-1B-CLM [76], SaProt-650M-AF2 [77], ESM-IF1 [78], Unirep evotuned [79], S3F-MSA [80], MSA-Transformer ensemble [81], VespaG [82], SiteRM [83]). Four lightweight Cα contact-map features are appended: the number of residues within 8 Å, the mean residue distance, the local density within 10 Å, and an inverse-contact RSA proxy. Within each cross-validation fold, contact features are z-scored using only training statistics, missing extra-zero-shot values are filled with the per-fold training-set median, and the resulting 23-dimensional scalar block is standardised with training-fold mean and standard deviation. GP regression targets are quantile-normalised to a standard normal on the training fold via van der Waerden scores: rank r_i (1-indexed, average-tied) is mapped to $\Phi^{-1}\big((r_i - 0.5)/N\big)$, keeping quantiles strictly inside $(0, 1)$ and avoiding $\pm\infty$; test-time MSE is computed against z-scored targets to match the ProteinGym/Kermut benchmark convention, with predictions clipped to $[-10, 10]$ to guard against GP extrapolation blow-up on out-of-distribution folds.

Structure kernel. AutoScientists-Kermut adopts the Kermut structure kernel. For each variant indexed by (mutated residue position, mutant amino acid) AutoScientists-Kermut precomputes three pairwise residue matrices on the training fold: (i) a Hellinger distance between the full 20-amino-acid ProteinMPNN conditional-probability distributions at the mutated positions, (ii) an L1 distance in log-space, $|\log p(\text{aa}_i \mid \text{ctx}_i) - \log p(\text{aa}_j \mid \text{ctx}_j)|$, i.e. the absolute difference of the scalar log-probabilities at the respective mutant amino acids, and (iii) the Euclidean distance between their Cα coordinates. The structure kernel is the radial product $k_{\text{struct}}(i, j) = \exp(-\ell_h h_{ij} - \ell_p p_{ij} - \ell_d d_{ij})$ with positive-constrained per-component lengthscales, wrapped in a `ScaleKernel` whose output scale carries a $\text{LogNormal}(0, 0.5)$ prior.

Three-GP ensemble. Each ensemble member combines the structure kernel with a sequence kernel via a learnable mixture $k = \pi\, k_{\text{struct}} + (1 - \pi)\, k_{\text{seq}}$, where $\pi = \sigma(\text{raw\_}\pi)$ is reparameterised through a sigmoid. The mean function is a single-input `LinearMean` over the ESM-2 zero-shot score, providing a strong unsupervised prior on top of which the GP models residuals. The three members differ only in k_{seq} and in the sequence-side feature set: **(GP1)** a Matérn-3/2 ARD kernel on the $1280 + 23 = 1303$-dimensional concatenation of ESM-2 embedding and standardised scalars; **(GP2)** a Matérn-5/2 ARD kernel on the 1280-d ESM-2 embedding alone, providing a smoother sequence kernel that does not see the auxiliary scalars; **(GP3)** a linear kernel on a compact 18-d feature vector

formed by dropping the five trailing extra zero-shot predictors (ProSST-4096, S3F-MSA [80], MSA-Transformer ensemble [81], VespaG, and SiteRM) from the scalar block; the three log-probability scalars, the ESM-2 zero-shot score, the first ten extra zero-shot predictors, and all four contact-map features are retained ($23 - 5 = 18$). Predictions from members that converge are averaged; if a member fails for numerical reasons (e.g. Cholesky errors on a degenerate fold) it is silently dropped and the remaining members are averaged.

Likelihood and region-aware noise. AUTOSCIENTISTS-Kermut uses a `FixedNoiseGaussianLikelihood` with learnable additional homoscedastic noise carrying a $\mathrm{HalfCauchy}(0.3)$ prior. The fixed per-sample noise variances are derived from a region-aware schedule that down-weights training points whose CV-fold index is far from the held-out fold: with $b_i = |\mathrm{fold}_i - \mathrm{test\_fold}|$, a sample weight $w_i = 1 + 0.5\, e^{-b_i}$ is converted to fixed noise $\sigma_i^2 = \mathrm{clip}(\, 0.05\, (w_i^{-1} - \min_j w_j^{-1}),\, 10^{-4}, \infty)$. This places near-zero fixed noise on the most reliable points (those adjacent to the test fold) and inflates noise on points distant from the test region, encouraging the GP to fit local structure rather than distant-region trends.

Optimisation. Each GP's parameters (sequence-kernel ARD lengthscales, structure-kernel component lengthscales, structure outputscale, π, mean parameters, and additional likelihood noise) are jointly trained for 1000 steps of AdamW with learning-rate 10^{-1} decayed cosine-style to 10^{-3}, maximising the exact GP marginal likelihood. The same number of steps and the same optimiser are used for every protein and every CV fold, with random seeds fixed to 2024.

Cross-validation and reporting. AUTOSCIENTISTS-Kermut trains and evaluates on the 5-fold CV columns provided by the ProteinGym single-substitutions benchmark (`fold_random_5`, `fold_modulo_5`, or `fold_contiguous_5`), refitting all per-fold preprocessing, per-fold ensemble members, and per-fold target normalisation independently for each held-out fold.

G.2 Ablations of AUTOSCIENTISTS-Kermut

We ablate three key design choices that AUTOSCIENTISTS-Kermut introduces to the original Kermut. All runs use the same five-fold cross-validation on the SARS2-Spike binding task dataset. The ablations are as follows: (1) **No ensemble** keeps only GP1 (full ESM-2 + scalar features, Matérn-3/2 + ARD). GP2 (ESM-2 only, Matérn-5/2) and GP3 (compact 18-d, linear) are removed. (2) **No quantile-norm targets** trains on simple per-fold z-scored targets instead of mapping training targets to a standard normal via rank quantiles. (3) **No extra zero-shot scores** drops the 15 auxiliary zero-shot predictors (MIF, VenusREM, ProSST-{128,2048,4096}, RSALOR, ESCOTT, xTrimoPGLM-1B-CLM, SaProt_650M_AF2, ESM-IF1, Unirep_evotune, S3F_MSA, MSA_Transformer_ensemble, VespaG, SiteRM); only ESM-2 650M zero-shot remains in the scalar block.

Table S9: Ablations on AUTOSCIENTISTS-Kermut on ProteinGym peformance on `SPIKE_SARS2_Starr_2020_binding`. Per-split Spearman ↑ and MSE ↓ are reported. The best value is shown in **bold** and the second best is shown in *italics*.

	Contiguous		Modulo		Random		Average	
Variant	ρ	MSE	ρ	MSE	ρ	MSE	ρ	MSE
Kermut	0.6950	0.5830	0.7060	0.5220	0.8410	*0.2070*	0.7473	*0.4373*
AUTOSCIENTISTS	**0.7842**	*0.5229*	**0.8174**	*0.5004*	*0.9204*	0.3347	**0.8407**	0.4527
Ablations								
No ensemble	0.7455	0.6092	*0.8030*	0.5460	**0.9207**	0.3527	*0.8231*	0.5026
No quantile-norm	*0.7638*	**0.4595**	0.7782	**0.4263**	0.8791	**0.1645**	0.8070	**0.3501**
No extra ZS	0.7591	0.5802	0.7821	0.5718	0.9109	0.3590	0.8174	0.5037

The ablations in Table S9 show these three components that each contribute meaningfully to AUTOSCIENTISTS-Kermut's ranking performance. The ensemble is the single largest driver of Spearman correlation, with removing it causing the steepest average ρ drop (0.8407 to 0.8231). The three GPs also carry complementary inductive biases and their diversity is most valuable on the harder contiguous split where sequence extrapolation matters most. Quantile-normalising the training targets produces the second-largest ρ degradation (0.8407 to 0.8070) and is the dominant ablation across all three split types. Conversely, removing it improves MSE, since z-scored targets compress prediction

scale. The consistent Spearman drop confirms that mapping skewed DMS scores to a standard normal before fitting the GP is critical for learning a well-calibrated ranking function. The 15 auxiliary zero-shot predictors contribute a consistent and reliable gain in both ρ and MSE across all splits, with no single split driving the effect, suggesting they supply diverse evolutionary signal that the ESM-2 embedding alone does not capture rather than overfitting to a particular data regime.