RIGOR: REPRESENTATION-INSTRUMENTED GENERATION WITH OUTCOME-GATED REFINEMENT FOR BUDGETED DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

AI scientist systems built on large language models (LLMs) can generate hypotheses and design experiments, but evaluating a candidate can be costly in time, computation, or materials, so the choice of the next experiment matters. We present **R**epresentation-**I**nstrumented **G**eneration with **O**utcome-Gated **R**efinement (RIGOR), a framework for budgeted discovery that describes candidate designs with an explicit rubric of interpretable dimensions, which record how designs differ rather than how good they are. Measuring a candidate on these dimensions gives its rubric-based representation, which hypothesis generation and outcome prediction share. An LLM outcome predictor estimates each candidate's outcome and uncertainty from its complete design, its rubric-based representation, and previous outcomes, and an upper confidence bound (UCB) rule selects the next experiment. As evidence accumulates, RIGOR may propose new rubric dimensions, but it activates a dimension only if it reduces prediction error by at least 5% on the next five outcomes, with every forecast recorded before the outcome is observed. Across four scientific-design tasks and four MLS-Bench algorithm-design tasks, with Opus 4.7 and GPT-5.5, a budget of 24 evaluations, and three runs per setting, RIGOR outperforms existing baselines: it achieves the highest mean final best score in 11 of 16 model–task settings, the highest mean evaluated outcome in all 16, and the lowest within-run variation over the second half of the budget in 15. Its prospective gate admits only 16 of 86 proposed dimensions, each of which improves prediction of subsequent outcomes.

1 INTRODUCTION

Large language models (LLMs) increasingly support scientific and algorithmic discovery by proposing hypotheses, designing experiments, and improving candidate solutions through external evaluation (Boiko et al., 2023; Bran et al., 2024; Lu et al., 2026; Gottweis et al., 2026; Romera-Paredes et al., 2023; Novikov et al., 2025). When each evaluation requires substantial computation, laboratory time, or materials, discovery also depends on deciding which candidates to test.

The decision is difficult because a candidate design can combine numerical parameters, categorical mechanisms, and structural choices written in natural language. Bayesian optimization addresses expensive experiment selection using predicted outcomes and uncertainty (Srinivas et al., 2012), and recent work incorporates LLMs into prediction and optimization over complex design spaces (Ramos et al., 2026; Liu et al., 2024). Applying such prediction to free-form designs requires a representation: a discovery system must choose which aspects of a design to represent and how to revise this representation as experimental evidence accumulates.

We introduce **R**epresentation-**I**nstrumented **G**eneration with **O**utcome-Gated **R**efinement (RIGOR), a framework for budgeted discovery built around an explicit, evolving representation of candidate designs. The representation is defined by a *rubric*: a set of interpretable dimensions that describe how candidate designs differ rather than how good they are. Measuring a candidate on these dimensions yields its *rubric-based representation*, or *profile* for short. Hypothesis generation and outcome prediction share this representation, and the complete design text stays available to both. The name reflects the two ideas of the method: generation and selection are instrumented with the

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

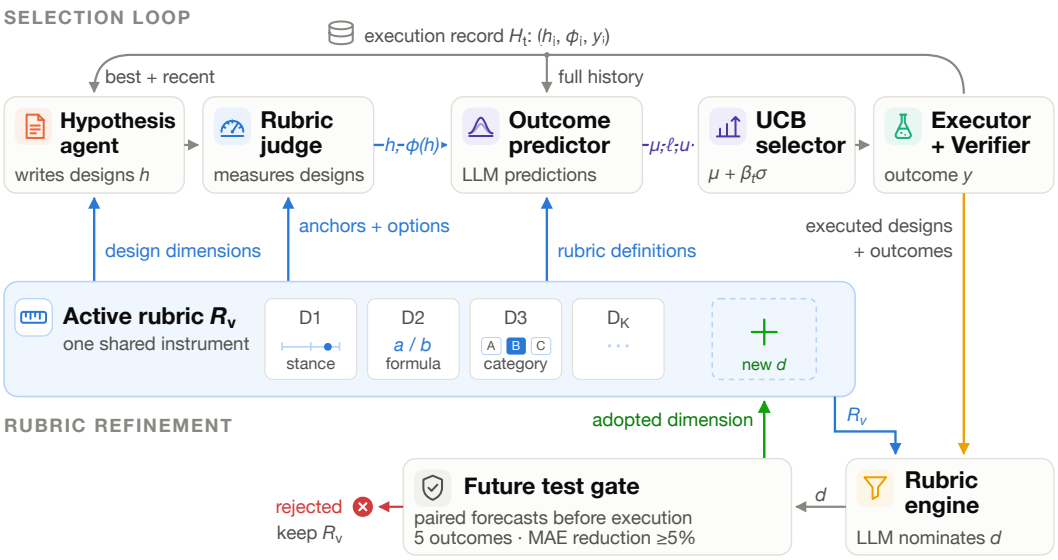

Figure 1: **RIGOR overview.** The hypothesis agent writes candidate designs h; the rubric judge measures the profile $\phi(h)$ of each design, which is its rubric-based representation; and the outcome predictor reads the complete design and its profile to produce the prediction and uncertainty that UCB selection uses. Experimental outcomes update the execution history. The rubric engine, an LLM reflection step, nominates missing dimensions. A nominated dimension stays inactive while paired forecasts with and without it are evaluated on subsequent outcomes, and it enters generation and selection only if it passes this future test (the prospective gate).

rubric-based representation of every candidate, and experimental outcomes gate each refinement of that representation.

RIGOR uses this representation throughout the discovery loop (Figure 1). At each round, it generates candidate designs, measures them on the active rubric, and predicts each candidate's outcome and uncertainty from its complete design, its rubric-based representation, and prior experiments. An upper confidence bound (UCB) rule selects the next experiment. As evidence accumulates, an LLM reflection step may propose a dimension that the rubric lacks. A proposed dimension does not affect generation or selection until it passes a prospective test.

Separating *proposing* a design distinction from *activating* it guards against representations that explain past outcomes without predicting future ones. For the next five successful experiments after a proposal, forecasts under the active rubric and under the rubric extended with the proposal are recorded before each outcome is observed, and the dimension is activated only if the extended forecasts reduce the mean absolute error (MAE) by at least 5%. Experimental outcomes decide both which candidates the system tests and which design distinctions it keeps.

Related methods also impose structure on LLM-guided search. PiEvo evolves scientific principles for hypothesis selection (Pu et al., 2026), while ReElicit learns semantic features for Bayesian optimization and generates candidates from acquisition-selected targets (Lin et al., 2026). RIGOR differs from both in two ways: its predictor reads each complete design together with its rubric profile, so selected distinctions become explicit without compressing the design into a feature vector, and a new rubric dimension becomes active only after it improves forecasts of later outcomes.

We evaluate RIGOR on four scientific-design tasks used by PiEvo (Pu et al., 2026) and four algorithm-design tasks from MLS-Bench (Lyu et al., 2026). We compare it with PiEvo and ReElicit under a budget of 24 evaluations, with Opus 4.7 and GPT-5.5 and three independent runs per method and model–task setting (144 runs in total). Across 16 settings, RIGOR achieves the highest mean final best score in 11, the highest mean evaluated outcome in all 16, and the lowest within-run variation over the second half of the budget in 15. Its outcome predictor has lower MAE than

the median of past outcomes in 46 of 48 runs. Prospective refinement is selective: only 16 of 86 proposed dimensions are adopted, and each of them improves prediction of subsequent outcomes.

Our contributions are:

- **A shared, rubric-based representation for discovery.** Hypothesis generation and outcome prediction share an interpretable, rubric-based representation of design choices. Its stance, formula, and categorical dimensions record design commitments, computed quantities, and mechanism choices in one form, and an LLM judge measures them from quoted evidence without access to outcomes.
- **Prospectively validated refinement.** An outcome-gated rule that activates a proposed rubric dimension only if it improves forecasts of outcomes observed after the proposal. Fitting past results is therefore not enough for a dimension to enter the rubric, and the test needs no additional benchmark evaluations.
- **Evaluation under fixed budgets.** A comparison with PiEvo and ReElicit on eight design tasks with two foundation models (144 runs of 24 evaluations each), together with an analysis of all 86 prospective refinement windows. Beyond the final best score, the evaluation measures the quality of every evaluated design, the accuracy and calibration of the outcome predictor, the effect of uncertainty on selection, and the stability of gate decisions under other thresholds and window lengths.

2 Related work

LLM-guided scientific and algorithmic discovery. LLM-based systems automate parts of scientific workflows through tool use, hypothesis generation, experiment design, and iterative evaluation (Boiko et al., 2023; Bran et al., 2024; Lu et al., 2026; Gottweis et al., 2026; Roohani et al., 2024). Program-search systems such as FunSearch and AlphaEvolve improve executable artifacts using feedback from external evaluators (Romera-Paredes et al., 2023; Novikov et al., 2025). PiEvo searches over evolving scientific principles with Gaussian-process-based selection (Pu et al., 2026). Search in RIGOR is organized around design choices rather than principles: a measured rubric-based representation structures both hypothesis generation and outcome prediction.

Language models, Bayesian optimization, and representations. LLMs have served as iterative optimizers (Yang et al., 2023), as in-context outcome predictors and uncertainty-aware experiment optimizers (Ramos et al., 2026; Ranković et al., 2026), and as Bayesian-optimization components for surrogate modeling, candidate generation, and search (Liu et al., 2024; Agarwal et al., 2025). Scientific Bayesian optimization has also incorporated explicit hypotheses and adaptive molecular representations (Cissé et al., 2024; Rajabi-Kochi et al., 2025). ReElicit learns semantic features from evaluated prompts, fits a Gaussian process in that space, and generates candidates from acquisition-selected feature targets (Lin et al., 2026). RIGOR does not compress designs into a learned feature space: its predictor reads the complete design text together with the rubric measurements, and a UCB rule selects experiments (Srinivas et al., 2012). LLM feature-engineering methods select generated features by their predictive value (Hollmann et al., 2023; Abhyankar et al., 2025), whereas RIGOR freezes each proposed dimension and tests it on a prespecified window of later outcomes before activating it.

Rubric-based evaluation and adaptation. LLM judges evaluate open-ended outputs at scale (Zheng et al., 2023; Liu et al., 2023), and LLM-Rubric decomposes such evaluations into explicit dimensions (Hashemi et al., 2024). Evidence-grounded methods tie judgments to textual support (Hong et al., 2026). Other work uses rubrics as reward signals or adapts them from comparisons and training feedback (Gunjal et al., 2025; Rezaei et al., 2025; Guan et al., 2026; Shao et al., 2025). RIGOR uses a rubric to describe design choices, not quality: the judge never observes experimental outcomes, and only the outcome predictor models performance. After initialization, the rubric changes only when a proposed dimension passes the prospective test, not through comparisons or training feedback.

3 METHOD

RIGOR maintains an explicit rubric-based representation of candidate designs, uses it for hypothesis generation and experiment selection, and updates it only when a proposed refinement improves prediction on future outcomes.

3.1 BUDGETED HYPOTHESIS SEARCH

Let h denote a complete design expressed in text and $y = f(h)$ its scalar outcome under a task-specific evaluator. We formulate all tasks as maximization; minimization scores are negated. Given an evaluation budget T, the goal is to maximize the best observed outcome $y_T^\star = \max_{1 \le i \le T} y_i$.

After t evaluations, the search state contains the execution history $\mathcal{H}_t = \{(h_i, y_i)\}_{i=1}^t$, an active rubric $\mathcal{R}_v$, the corresponding measured profiles, and any pending rubric proposal. RIGOR has three decision components: a hypothesis agent, a rubric judge, and an LLM outcome predictor; a reflection step proposes rubric refinements (Section 3.5). A task-specific Executor and Verifier produce outcomes, while a fidelity audit checks that executed implementations match their declared designs (Appendix C).

For code tasks, whose designs are implemented as programs, all components receive the same task context (the research question, the editable program, and its fixed calling code); hidden targets and scoring data remain accessible only to the Verifier.

3.2 RUBRIC-BASED REPRESENTATION

The active rubric $\mathcal{R}_v = (d_1, \ldots, d_K)$ is an ordered list of interpretable design dimensions, and v indexes rubric versions. Measuring a design h on these dimensions yields its profile $\phi_v(h)$ (Eq. 2 in Appendix B), which serves as its rubric-based representation. The complete design text is retained alongside the profile.

At initialization, several rubric drafts written independently by an LLM are merged into an initial rubric of K dimensions. To favor dimensions that capture design variation rather than quality, each draft must give, for every proposed dimension, a counterexample: a design that scores high on the dimension yet performs poorly overall.

We use three types of dimensions. **Stance dimensions** represent opposing design commitments, with anchors at 0 and 10 and a neutral value of 5. **Formula dimensions** represent quantities computed from explicitly stated design parameters, with anchors defining ordered numerical ranges. **Categorical dimensions** record a choice among named mechanisms or "other", with no order among the options.

For stance and formula dimensions, the rubric judge classifies whether a design engages the dimension, in which direction, and how strongly; a fixed map converts this closed classification to one of seven tier values between 0 and 10. For a categorical dimension, the judge selects one of the predefined options. Measurements are grounded in quoted evidence from the candidate text. We use J independent judge draws per candidate: tier-valued dimensions take the mean of valid draws, while categorical dimensions use the majority choice. The judge sees the candidate text and rubric calibration examples, but no outcome labels. Appendix B gives the full measurement protocol.

3.3 HYPOTHESIS GENERATION

At round t, the hypothesis agent proposes a slate $\mathcal{C}_t$ of M complete candidate designs using the task context, active rubric, and an execution memory of previously evaluated designs, outcomes, and profiles.

Once outcomes are available, the agent may declare up to M_c candidates per slate as *single-change children*. Each child names an evaluated parent, keeps the parent's complete specification, and declares exactly one modification; the other candidates are specified independently.

For code tasks, the Executor implements a child by editing its parent's evaluated code, so each child's outcome is tied to a measured parent and one declared modification. All candidates otherwise follow the same prediction, selection, and evaluation procedure.

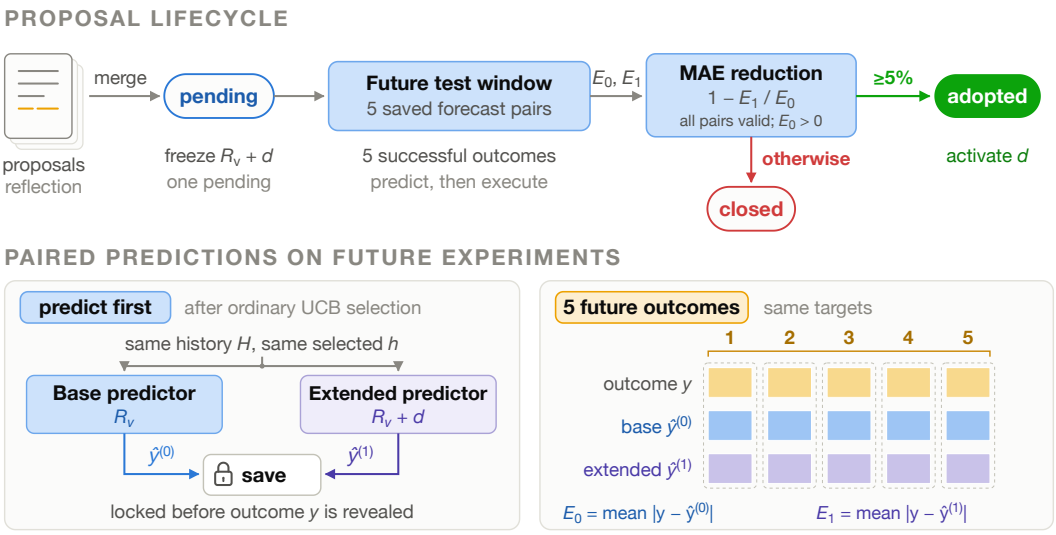

Figure 2: **Prospective rubric refinement.** A proposed dimension d is frozen and tested on W subsequent outcomes. For each selected design, two forecasts are recorded before execution from the same history: one with the active rubric $\mathcal{R}_v$ and one with the extended rubric $\mathcal{R}_v \cup \{d\}$. The dimension is adopted only if all forecast pairs are valid, the base MAE is positive, and the extended rubric reduces MAE by at least ρ. The figure shows the setting of all experiments, $W = 5$ and $\rho = 5\%$ (Section 4.1).

3.4 Outcome Prediction and Experiment Selection

For each candidate $h \in \mathcal{C}_t$, the LLM outcome predictor conditions on the complete design, its rubric-based representation, and the history of executed designs and outcomes. It returns a point prediction and a central interval with nominal coverage $1 - \alpha$.

We make S predictor draws and take componentwise medians of their point predictions and interval endpoints, yielding $\hat{\mu}_t(h)$, $\hat{\ell}_t(h)$, and $\hat{u}_t(h)$. The interval width defines a Gaussian-equivalent uncertainty scale $\hat{\sigma}_t(h) = (\hat{u}_t(h) - \hat{\ell}_t(h))/(2z_{1-\alpha/2})$, where $z_{1-\alpha/2}$ is the standard normal quantile. The next experiment maximizes the upper confidence bound $a_t(h) = \hat{\mu}_t(h) + \beta_t \hat{\sigma}_t(h)$ over the slate, $h_{t+1} = \arg\max_{h \in \mathcal{C}_t} a_t(h)$, with $\beta_t = \beta_0(1 - t/T)$. Because β_t decreases linearly with t, selection shifts from exploration toward exploitation as the budget is consumed. Appendix B gives further prediction details.

3.5 Prospective Rubric Refinement

Proposal and activation are separate steps: a dimension is proposed from past outcomes but activated only after it improves forecasts of later ones (Figure 2).

Nomination and freezing. After at least $t_{\min}$ outcomes, an LLM reflection step reviews the execution history, the parent–child comparisons, and the active rubric, and may nominate one design distinction that the rubric lacks. A nomination equivalent to an earlier proposal is merged with it. For a new proposal d, the specification of d and the base rubric are frozen, and d is measured on the existing history. Only one proposal may be pending at a time, and at most $A_{\max}$ dimensions may be adopted per run.

Prospective predictions. Suppose d is proposed after t_0 evaluations. For each of the next W successful experiments, UCB selection continues to use the active rubric $\mathcal{R}_v$. After selecting h_{t_0+i} but before observing its outcome, RIGOR records $\hat{y}_i^{(0)} = P(\mathcal{R}_v, \mathcal{H}_{t_0+i-1}, h_{t_0+i})$ and $\hat{y}_i^{(1)} = P(\mathcal{R}_v \cup \{d\}, \mathcal{H}_{t_0+i-1}, h_{t_0+i})$, where P is the same median-of-S predictor used for selection.

The two forecasts differ only in whether d and its measurements are included; the history grows as outcomes arrive, while the base rubric and d stay fixed, and neither d nor its forecasts affect generation or selection while the test is pending.

Decision and activation. After W successful outcomes, we compute the paired mean absolute errors $E_0 = W^{-1}\sum_{i=1}^{W}|y_{t_0+i} - \hat{y}_i^{(0)}|$ and $E_1 = W^{-1}\sum_{i=1}^{W}|y_{t_0+i} - \hat{y}_i^{(1)}|$. The proposal is adopted if and only if all W forecast pairs are valid, $E_0 > 0$, and $E_1 \leq (1-\rho)E_0$.

If adopted, d is appended to the active rubric and becomes available to generation, measurement, and outcome prediction; otherwise the rubric is unchanged. The test uses no benchmark evaluations beyond those of the search itself. Algorithm 1 in Appendix A gives the complete schedule.

4 Experiments

We evaluate RIGOR on eight design tasks with two foundation models and ask four questions. (1) Does RIGOR find better designs within a fixed evaluation budget? (2) Does it evaluate strong candidates throughout the budget? (3) Are its outcome predictions and uncertainty informative for experiment selection? (4) How selective and stable is prospective rubric refinement?

4.1 Experimental setup

Tasks. We use four scientific-design tasks from PiEvo (Pu et al., 2026), namely Nanohelix, ChEMBL, SuperCon, and transition-metal complexes (TMC), and four algorithm-design tasks from MLS-Bench (Lyu et al., 2026), namely clustering, online bandits, scaling-law modeling, and multi-objective evolutionary algorithms (MOEA). The tasks span continuous parameters, molecular and material structures, categorical mechanisms, and executable algorithms. Table 4 in Appendix C gives the design space and objective for each task.

Methods and budget. We compare RIGOR with the official PiEvo implementation and our implementation of ReElicit (Lin et al., 2026). Each method runs with Opus 4.7 and with GPT-5.5, and all methods share the task description, evaluator, feasibility constraints, and budget of 24 evaluations. One evaluation is one complete experiment submitted to the evaluator. The two models and eight tasks give 16 model–task settings; with three independent runs per method and setting, the comparison comprises 144 runs and 3,456 evaluations. ReElicit spends the same 24 evaluations as six batches of four. Appendix C gives the full protocols and budget accounting.

Implementation. All RIGOR runs use the same configuration across tasks and models: $M = 6$ candidates per slate, $J = 3$ judge draws, $S = 3$ predictor draws with 80% intervals, $\beta_0 = 1.5$, and prospective windows of $W = 5$ outcomes with adoption threshold $\rho = 5\%$. Table 3 in Appendix C lists all algorithmic hyperparameters.

Metrics. Let $B_t = \max_{i \leq t} y_i$ be the incumbent, the best outcome after t evaluations; our primary metric is the final best score B_{24}. All statistics are computed within each run and then averaged across the three runs of a setting. Figures show the mean and min–max range across runs.

4.2 Main results

RIGOR has the highest mean final best score in 11 of 16 settings. RIGOR leads in six of eight settings with Opus 4.7 and five of eight with GPT-5.5 (Table 1 and Figure 3). It leads on Nanohelix and TMC with both models, on ChEMBL with Opus 4.7, and in six of the eight MLS-Bench settings. On TMC, its mean final best score exceeds the stronger baseline by 7.0% with Opus 4.7 and 10.0% with GPT-5.5. In the remaining five settings, RIGOR is within 4.5% of the leading method, and it ranks first or second in 14 of 16 settings (Appendix E). Its mean rank is 1.44, compared with 2.06 for PiEvo and 2.50 for ReElicit, and its average gap to the best mean is 0.8%, compared with 3.5% and 6.5%.

These leads span different kinds of design spaces: numerical geometry, molecular and ligand identities, exploration schedules, functional forms, and algorithmic mechanisms. One RIGOR configu-

Table 1: Final best score B_{24} after 24 evaluations: mean over three runs, with sample standard deviation in gray. Higher is better. Boldface marks the best mean in each setting; RIGOR columns are shaded.

	Opus 4.7			GPT-5.5		
Task	RIGOR	PiEvo	ReElicit	RIGOR	PiEvo	ReElicit
Scientific design						
Nanohelix	**1.101** ± 0.001	1.040 ± 0.039	1.027 ± 0.045	**1.083** ± 0.017	1.068 ± 0.009	0.949 ± 0.027
ChEMBL	**5.061** ± 0.220	4.868 ± 0.146	4.779 ± 0.040	4.887 ± 0.085	4.901 ± 0.117	**5.115** ± 0.115
SuperCon	109.3 ± 0.1	**112.8** ± 2.6	110.2 ± 1.0	111.2 ± 1.0	**114.0** ± 2.0	110.3 ± 1.0
TMC	**482.0** ± 21.5	438.4 ± 17.3	450.5 ± 13.8	**477.8** ± 30.5	434.4 ± 12.9	432.9 ± 56.3
MLS-Bench						
Clustering	**0.577** ± 0.006	0.552 ± 0.015	0.557 ± 0.005	0.553 ± 0.006	0.552 ± 0.008	**0.562** ± 0.008
Online bandit	**0.805** ± 0.064	0.763 ± 0.122	0.759 ± 0.153	**0.975** ± 0.022	0.942 ± 0.101	0.896 ± 0.056
Scaling law	**0.550** ± 0.017	0.542 ± 0.010	0.487 ± 0.032	**0.605** ± 0.033	0.569 ± 0.053	0.583 ± 0.016
MOEA	0.496 ± 0.011	**0.501** ± 0.017	0.442 ± 0.077	**0.509** ± 0.007	0.504 ± 0.010	0.441 ± 0.069

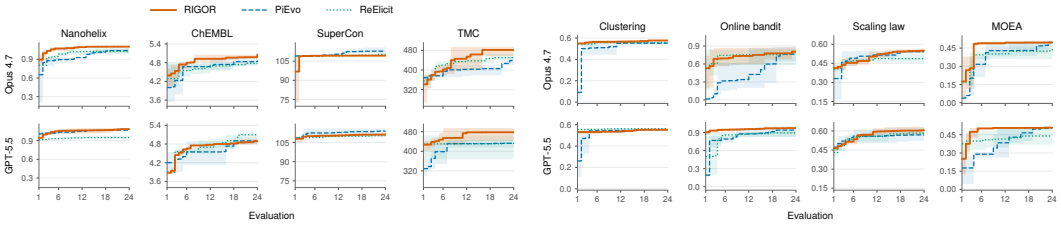

Figure 3: **Best-so-far score over 24 evaluations.** Scientific-design tasks (left) and MLS-Bench tasks (right). Lines show the mean over three runs; shaded regions span the minimum and maximum. Appendix D.1 gives enlarged versions.

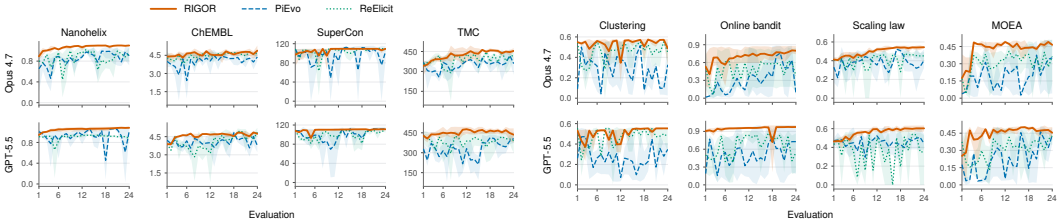

Figure 4: **Outcome of the design evaluated at each step.** Scientific-design tasks (left) and MLS-Bench tasks (right). Lines show the mean over three runs; shaded regions span the minimum and maximum. RIGOR has the highest mean outcome in 320 of 384 model–task–step comparisons (83.3%). Appendix D.1 gives enlarged versions.

ration serves all of them: the rubric-based representation records numerical, categorical, and stance decisions in one form, and the predictor still reads the complete design.

4.3 Search quality across the budget

To measure how efficiently each run uses its full budget, we also compute the mean incumbent $A = \frac{1}{24}\sum_{t=1}^{24} B_t$, the mean evaluated outcome $Q = \frac{1}{24}\sum_{t=1}^{24} y_t$, and, over evaluations 13–24, the late mean Q_{late} and the late within-run standard deviation S_{late}.

RIGOR evaluates stronger candidates throughout the budget. RIGOR has the highest mean evaluated outcome Q in all 16 settings and the highest late mean Q_{late} in all 16 (Table 2). It also has the lowest late within-run variation in 15 settings. Across all 384 model–task–step comparisons,

Table 2: Number of settings, out of 16, in which each method has the best mean statistic. Statistics are computed within each run and averaged over three runs. Appendix D reports all values.

Statistic	RIGOR	PiEvo	ReElicit
Final best score B_{24}	**11/16**	3/16	2/16
Mean incumbent A	**11/16**	2/16	3/16
Mean evaluated outcome Q	**16/16**	0/16	0/16
Late mean outcome Q_{late} (evaluations 13–24)	**16/16**	0/16	0/16
Late within-run variation S_{late} (lower is better)	**15/16**	0/16	1/16

RIGOR has the highest mean outcome in 320 (83.3%; Figure 4). For example, on clustering with Opus 4.7, Q_{late} is 0.552 for RIGOR, compared with 0.257 for PiEvo and 0.490 for ReElicit; the corresponding S_{late} values are 0.035, 0.184, and 0.063.

Later evaluations are stronger and more concentrated than early ones. All 48 RIGOR runs improve on the best of their first four evaluations, and 42 reach their final best score after evaluation 12. In 46 runs, the mean outcome of the second half exceeds that of the first half, and in 44 the second half varies less. We attribute this stability to the rubric-based representation that the hypothesis agent and the predictor share: both refer to the same named design decisions, and the execution memory attaches these decisions to measured outcomes.

4.4 Outcome prediction and experiment selection

The outcome predictor is more accurate than a history-only reference. We compare the forecast saved before each of the 1,152 RIGOR evaluations with a reference forecast, the median of the earlier outcomes in the same run. Over evaluations 2–24, the predictor has lower MAE in 46 of 48 runs, and the mean ratio of its MAE to the reference MAE is 0.621 (Appendix F). It is also more accurate than the mean of the earlier outcomes, the mean of the last three, and the last outcome, each in at least 43 of 48 runs. When the predictor forecasts that the selected candidate will beat the incumbent, it does so in 52.7% of cases, compared with 24.4% otherwise.

Predicted uncertainty tracks forecast error. Within a run, the width of the prediction interval has a median Spearman correlation of 0.55 with the absolute forecast error, and the correlation is positive in 43 of 48 runs. Coverage of the nominal 80% intervals increases from 69.8% in evaluations 1–8 to 78.1% in evaluations 9–16 and 81.2% in evaluations 17–24.

UCB becomes less exploratory as the budget is consumed. In 248 of 1,152 selections (21.5%), UCB chooses a candidate whose point forecast is below the maximum point forecast in its slate, so uncertainty changes the selected experiment. This occurs in 32.0% of evaluations 1–8, 26.6% of evaluations 9–16, and 6.0% of evaluations 17–24, matching the decay of β_t.

Single-change children and independent candidates both improve the incumbent. Of the 1,152 executed designs, 686 are single-change children of an evaluated parent. After the first evaluation of each run, the incumbent improves 375 times: 202 improvements come from single-change children and 173 from independently specified candidates. For all 361 single-change children on MLS-Bench, the Executor starts from the evaluated implementation of the parent, so each child is a controlled edit of measured code.

4.5 Prospective rubric refinement

Across the 48 RIGOR runs, the gate completes 86 five-outcome windows, comprising 430 paired forecasts. Both forecasts of every pair are recorded before the outcome is observed, from the same history, with the base rubric and the proposed dimension fixed for the whole window.

Only 16 of 86 proposed dimensions are adopted. The 16 adopted dimensions reduce MAE by 5.38–41.80% over their five-outcome windows (Appendix G). They capture concrete design choices

Case 1: From an adopted dimension to a code edit — Online bandit, GPT-5.5

Prospective gate		*MAE reduction*
Evals. 5–9: adopted	*Contextual forced initial coverage per arm*	**31.29%**
Design trace		*Outcome*
Eval. 10	Eight forced initial selections per arm	0.941
Eval. 11	Child of 10: coverage threshold $8 \rightarrow 6$	**0.943**
Eval. 21	KL-UCB exploration coefficient 0.35	0.956
Eval. 22	Child of 21: coefficient $0.35 \rightarrow 0.30$	**0.964**

The first child refines the decision named by the adopted dimension, and the second gives the best result of the run. Each child changes exactly one line of its parent's code.

Figure 5: **A run traced from an adopted dimension to code edits.** A child declares one change to an evaluated parent design.

including molecular scaffold family, ligand-pair identity, contextual forced initial coverage in a bandit policy, and residual-feature construction in a scaling-law model.

The gate rejects the remaining 70 proposals: 52 increase prediction error, 5 leave it unchanged, and 13 reduce it by less than the required 5%. On clustering and MOEA, the gate rejects all 18 proposals.

Most adopted dimensions pass a stricter threshold. Eleven of the 16 adopted dimensions also pass a 10% MAE-reduction threshold (Appendix F). Most proposals are rejected at every threshold from 0% to 20%. Decisions also stabilize with longer windows: compared with the five-outcome decision, agreement is 63/86 after two outcomes and 78/86 after four. A two-outcome gate would adopt 14 dimensions whose gain does not persist over five outcomes.

4.6 Case study: from representation to code

Figure 5 traces an online-bandit run with GPT-5.5 in which an adopted dimension names a design decision and single-change children turn that decision into one-line edits of evaluated code. After evaluation 9, the gate adopts a dimension for contextual forced initial coverage per arm. Evaluation 11, a child of evaluation 10, lowers the coverage threshold from eight to six by changing one line of code, and the score rises from 0.941 to 0.943. A later one-line change to the KL-UCB coefficient raises the score from 0.956 to the run best of 0.964.

5 Conclusion

RIGOR uses a shared rubric-based representation of design choices to connect hypothesis generation, outcome prediction, and rubric refinement. Uncertainty-aware selection and independently specified candidates sustain exploration, while prospective paired forecasts determine when new distinctions enter the representation. Across 144 runs on eight tasks and two foundation models, RIGOR achieves the highest mean final best score in 11 of 16 settings, the highest mean evaluated outcome in all 16, and the lowest late within-run variation in 15. Its predictor beats a historical-median forecast in 46 of 48 runs, and its gate adopts only 16 of 86 proposed dimensions. These results support using experimental outcomes to guide both candidate selection and representation refinement when the evaluation budget is limited.

Limitations. RIGOR targets settings in which each evaluation is expensive; it uses LLM calls on measurement, prediction, and prospective testing before every experiment to improve efficiency and reduce the cost of running real experiments. When candidates can be validated in seconds and evaluation cost does not matter, high-throughput search can iterate faster. Even then, the rubric-based representation offers better interpretability by recording which design decisions predict outcomes.

AI USE STATEMENT

In this work, we used LLMs to assist with writing, including drafting and editing the text of the paper. All other parts of this work, including the research ideas, the design of RIGOR, and the experimental protocol, are our own. The foundation models that RIGOR and the baselines call during search are components of the evaluated systems and are described in Sections 3 and 4.1. We have reviewed all AI-assisted text and take responsibility for the final content of this work, including text produced with the aid of LLMs.

ETHICS STATEMENT

This work involves no human subjects, crowdsourced annotation, or personal data. All experiments are computational and use publicly available benchmarks: the scientific-design tasks score each candidate with a pretrained evaluator, and the MLS-Bench tasks run each implemented algorithm with the benchmark's official Verifier. No molecule, material, metal complex, or nanostructure was synthesized, and no physical experiment was performed. The objectives are the benchmark targets (chiroptical response, binding affinity of drug-like molecules, superconducting critical temperature, polarizability, and algorithm performance), and no task rewards toxicity or any other harmful property. If the discovery loop is connected to physical synthesis, each selected design should pass a laboratory safety review before it is made.

REPRODUCIBILITY STATEMENT

All tables and figures use the outcomes recorded in the 144 runs and the forecasts recorded before execution in the 48 RIGOR runs, with no additional model calls or experiments. Algorithm 1 specifies selection, paired prediction, execution, and rubric activation. Section 4.1, Table 3, and Appendix C provide the hyperparameters, task protocols, and budget accounting.

REFERENCES

Nikhil Abhyankar, P. Shojaee, and Chandan K. Reddy. LLM-FE: Automated Feature Engineering for Tabular Data with LLMs as Evolutionary Optimizers. *ArXiv*, abs/2503.14434, 2025.

Dhruv Agarwal, Manoj Ghuhan Arivazhagan, Rajarshi Das, Sandesh Swamy, Sopan Khosla, and Rashmi Gangadharaiah. Searching for Optimal Solutions with LLMs via Bayesian Optimization. In *International Conference on Learning Representations*, 2025.

Daniil A. Boiko, R. MacKnight, Benjamin C Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624:570–578, 2023.

Andrés M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and P. Schwaller. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6:525–535, 2024.

Abdoulatif Cissé, Xenophon Evangelopoulos, Sam Carruthers, V. Gusev, and A. Cooper. HypBO: Accelerating Black-Box Scientific Experiments Using Experts' Hypotheses. In *International Joint Conference on Artificial Intelligence*, pp. 3881–3889, 2024.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, Anil Palepu, Keran Rong, Ryutaro Tanno, Khaled Saab, Fan Zhang, Jacob Blum, Andrew Carroll, Kavita Kulkarni, Nenad Tomašev, Dina Zverinski, Ivor Rendulic, Elahe Vedadi, Florian Hasler, Luka Rimanic, Marina Boia, Ivan Budiselic, Ben Feinstein, Mathias Bellaiche, Tom Sheffer, Jan Freyberg, Jeremy Ratcliff, Ottavia Bertolli, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yossi Matias, James Manyika, Demis Hassabis, Yunhan Xu, Pushmeet Kohli, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Accelerating scientific discovery with Co-Scientist. *Nature*, 655:487–496, 2026.

Xin Guan, Xiaomeng Hu, Shen Huang, Zhenyi Wang, Bo Zhang, Zijian Li, Pengjun Xie, Bo Liu, and Jiuxin Cao. EvoRubric: Self-Evolving Rubric-Driven RL for Open-Ended Generation. *ArXiv*, abs/2605.29847, 2026.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as Rewards: Reinforcement Learning Beyond Verifiable Domains. *ArXiv*, abs/2507.17746, 2025.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. LLM-Rubric: A Multidimensional, Calibrated Approach to Automated Evaluation of Natural Language Texts. *ArXiv*, abs/2501.00274, 2024.

N. Hollmann, Samuel Müller, and F. Hutter. Large Language Models for Automated Data Science: Introducing CAAFE for Context-Aware Automated Feature Engineering. *Advances in Neural Information Processing Systems 36*, 2023.

Yihan Hong, Huaiyuan Yao, Bolin Shen, Wanpeng Xu, Hua Wei, and Yushun Dong. From Rubrics to Reliable Scores: Evidence-Grounded Text Evaluation with LLM Judges. 2026.

Zhiyuan Jerry Lin, Benjamin Letham, Samuel Dooley, Maximilian Balandat, and Eytan Bakshy. Embedding by Elicitation: Dynamic Representations for Bayesian Optimization of System Prompts. *ArXiv*, abs/2605.19093, 2026.

Tennison Liu, N. Astorga, Nabeel Seedat, and M. van der Schaar. Large Language Models to Enhance Bayesian Optimization. *ArXiv*, abs/2402.03921, 2024.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. *ArXiv*, abs/2303.16634, 2023.

Chris Lu, Cong Lu, R. Lange, Yutaro Yamada, Shengran Hu, J. Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of AI research. *Nature*, 651:914–919, 2026.

Bohan Lyu, Yucheng Yang, Siqiao Huang, Jiaru Zhang, Qixin Xu, Xinghan Li, Xinyang Han, Yicheng Zhang, Huaqing Zhang, Runhan Huang, Kaicheng Yang, Zitao Chen, Wentao Guo, Junlin Yang, Xinyue Ai, Wenhao Chai, Yadi Cao, Ziran Yang, Kun Wang, Dapeng Jiang, Huan-ang Gao, Shange Tang, Chengshuai Shi, Simon S. Du, Max Simchowitz, Jiantao Jiao, Dawn Song, and Chi Jin. MLS-Bench: A Holistic and Rigorous Assessment of AI Systems on Building Better AI. *ArXiv*, abs/2605.08678, 2026.

Alexander Novikov, Ngân Vu, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, S. Shirobokov, B. Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. P. Kumar, Abigail See, Swarat Chaudhuri, George Holland, A. Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. AlphaEvolve: A coding agent for scientific and algorithmic discovery. *ArXiv*, abs/2506.13131, 2025.

Yingming Pu, Tao Lin, and Hongyu Chen. Principle-Evolvable Scientific Discovery via Uncertainty Minimization. *ArXiv*, abs/2602.06448, 2026.

M. Rajabi-Kochi, Negareh Mahboubi, Aseem Partap Singh Gill, and S. Moosavi. Adaptive representation of molecules and materials in Bayesian optimization. *Chemical Science*, 2025.

Mayk Caldas Ramos, Shane S Michtavy, Andrew D. White, and Marc D. Porosoff. Bayesian Optimization of Catalysis with In-Context Learning. *ACS Central Science*, 12:599–615, 2026.

Bojana Ranković, Ryan-Rhys Griffiths, and P. Schwaller. Large language models as uncertainty-calibrated optimizers for experimental discovery. *Nature Machine Intelligence*, 8:1466–1477, 2026.

MohammadHossein Rezaei, Robert Vacareanu, Zihao Wang, Clinton Wang, Bing Liu, Yunzhong He, and Afra Feyza Akyürek. Online Rubrics Elicitation from Pairwise Comparisons. *ArXiv*, abs/2510.07284, 2025.

B. Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. P. Kumar, Emilien Dupont, Francisco J. R. Ruiz, J. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2023.

Yusuf H. Roohani, Andrew H. Lee, Qian Huang, Jian Vora, Zachary Steinhart, Ke-Xin Huang, A. Marson, Percy Liang, and J. Leskovec. BioDiscoveryAgent: An AI Agent for Designing Genetic Perturbation Experiments. *ArXiv*, abs/2405.17631, 2024.

Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, V. Kishore, Jingming Zhuo, Xin-Ran Zhao, Molly Park, S. Finlayson, David Sontag, Tyler Murray, Sewon Min, Pradeep Dasigi, Luca Soldaini, Faeze Brahman, Wen-tau Yih, Tongshuang Wu, Luke S. Zettlemoyer, Yoon Kim, Hanna Hajishirzi, and Pang Wei Koh. DR Tulu: Reinforcement Learning with Evolving Rubrics for Deep Research. *ArXiv*, abs/2511.19399, 2025.

Niranjan Srinivas, Andreas Krause, S. Kakade, and M. Seeger. Information-Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting. *IEEE Transactions on Information Theory*, 58:3250–3265, 2012.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large Language Models as Optimizers. *ArXiv*, abs/2309.03409, 2023.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *ArXiv*, abs/2306.05685, 2023.

A COMPLETE DISCOVERY LOOP

Algorithm 1 The RIGOR discovery loop, with at most one pending rubric proposal

Require: Task context q; hyperparameters T, W, ρ, β_0, $t_{\min}$, $A_{\max}$ (Table 3)

$\mathcal{R} \leftarrow \mathrm{AuthorRubric}(q)$; $\mathcal{H} \leftarrow \varnothing$; $b \leftarrow \varnothing$
$n \leftarrow 0$; $r \leftarrow -1$ ▷ adoptions; last reflected history size
while $|\mathcal{H}| < T$ **do** ▷ $|\mathcal{H}|$ counts submitted experiments
  $t \leftarrow |\mathcal{H}|$
  if $b = \varnothing$, $t \geq t_{\min}$, $t > r$, $T - t > W$, and $n < A_{\max}$ **then**
    $d \leftarrow \mathrm{Reflect}(q, \mathcal{R}, \mathcal{H})$; $r \leftarrow t$
    if d is a new, non-equivalent dimension **then**
      Freeze $b = (\mathcal{R}, d)$; measure d on historical designs
    end if
  end if
  $\mathcal{C} \leftarrow \mathrm{Generate}(q, \mathcal{R}, \mathcal{H})$ ▷ M complete designs
  Measure active profiles; retain candidates with a usable profile
  $(\hat{\mu}, \hat{\ell}, \hat{u}) \leftarrow \mathrm{PredictMedian}_S(q, \mathcal{R}, \mathcal{H}, \mathcal{C})$
  $\beta_t \leftarrow \beta_0(1 - t/T)$
  $h^\star \leftarrow \arg\max_{h \in \mathcal{C}} \hat{\mu}(h) + \beta_t(\hat{u}(h) - \hat{\ell}(h))/(2z_{1-\alpha/2})$
  if $b \neq \varnothing$ **then**
    Measure d on $h^\star$
    Save paired median-of-S forecasts under the frozen base and extended rubrics
  end if
  $(y, \mathrm{code}, \mathrm{status}) \leftarrow \mathrm{ExecuteVerifyAudit}(h^\star)$
  if status is not successful **then**
    $y \leftarrow 0$ ▷ an invalid design uses its unit and scores zero
  else if $b \neq \varnothing$ **then**
    Attach y to its saved forecast pair
  end if
  Append $(h^\star, y)$ to $\mathcal{H}$ and store its active profile
  if b has W successful outcome targets **then**
    decision $\leftarrow$ reject
    if all W forecast pairs are valid **then**
      Compute paired MAEs E_0 and E_1 on the same W outcomes
      if $E_0 > 0$ and $E_1 \leq (1 - \rho)E_0$ **then**
        $\mathcal{R} \leftarrow \mathcal{R} \cup \{d\}$; increment version; $n \leftarrow n + 1$
        Add the measured values of d to the profiles; decision $\leftarrow$ adopt
      end if
    end if
    Close b with decision; $b \leftarrow \varnothing$
  end if
end while
return the executed design with the highest outcome

Candidates without a usable profile are excluded before prediction. $\mathrm{PredictMedian}_S$ takes componentwise medians of S predictor draws over the complete design texts and rubric profiles. $\mathrm{ExecuteVerifyAudit}$ includes implementation, verification, fidelity auditing, and repair; a declared child starts from its evaluated parent's implementation. Each iteration submits one experiment and uses one unit of the budget, so $|\mathcal{H}|$ counts submitted experiments, and an invalid design receives an outcome of zero (Appendix C). A prospective window counts only successful outcomes; a window that is still incomplete when the budget ends closes without adoption.

A prospective window opens only when more than W evaluations remain, so an adopted dimension can affect a subsequent selection. A new proposal equivalent to a closed one inherits that proposal's recorded decision. A missing paired forecast for a successful outcome remains in the fixed window and prevents adoption; it is neither replaced nor reconstructed after execution. Gate decisions take effect before the next nomination and generation step.

B MEASUREMENT AND PREDICTOR DETAILS

Typed profiles. For stance and formula dimensions, the judge returns a closed classification of whether the design engages the dimension, in which direction, and how strongly. A fixed map converts this classification to a tier value:

	weak	medium	strong
positive pole	6.5	8	10
negative pole	3.5	2	0

$$s_{\text{neutral}} = 5. \tag{1}$$

Each candidate is judged with J independent draws. For tier-valued dimensions, the profile averages valid draws; categorical dimensions use the majority choice with a fixed option-order tie rule:

$$\phi_{v,k}(h) = \begin{cases} |V_k|^{-1} \sum_{j \in V_k} s_{j,k}(h), & d_k \text{ is tier-valued,} \\ \text{majority}_{j \in V_k}\, c_{j,k}(h), & d_k \text{ is categorical.} \end{cases} \tag{2}$$

Here V_k is the set of valid draws for dimension d_k. Profiles are shown to the agents with dimension names and category labels. A measurement that fails validation is requested again, and only candidates with a usable profile enter selection. Each categorical dimension has three to eight named options plus "other."

Evidence and calibration. The judge receives the candidate text, the task context, rubric anchors or options, and up to three previously measured calibration examples; calibration examples contain measurements but no outcome labels. Each asserted commitment or categorical choice must be supported by a quotation identifying a concrete parameter or operation. A quotation is accepted if it appears in the candidate text after whitespace, case, and typography are normalized. A design that addresses a stance dimension but takes a neutral position still needs a supporting quotation, whereas a design that does not mention the dimension receives the neutral tier. For a formula dimension, the judge evaluates the formula on the parameter values stated in the design and classifies where the result falls on the anchors; its quotation contains the input values, and a missing input yields a neutral measurement. The judge must correct any invalid classification or unsupported quotation before the measurement is accepted.

Generation context. The hypothesis agent receives the task and source context, active rubric definitions, categorical options or formulas, and the judge's instructions for classifying stances. Its execution memory includes the best design and the most recent distinct executed designs in full, with their outcomes and profiles; a compact list covers all executed designs, and, for each dimension, counts show how often each tier or option occurs among executed designs. The hypothesis agent chooses the mechanisms of all candidates freely; no target profile or prescribed scientific principle constrains the slate. With an empty history, it is prompted to cover the rubric space.

Prediction context. The predictor used for selection receives every executed design in order, with its outcome, active rubric measurements, and any parent comparison. The hypothesis agent receives the smaller memory described above. Both receive the task context. Pending-gate predictors use the same measured history and differ only in the proposed dimension and its measurements.

Selection. Before any outcomes are available, the predictor uses the task and design descriptions as prior information; the same UCB rule applies from the first selection. The exploration term comes from the width of the prediction interval; disagreement among judge draws serves only as a measurement diagnostic. Prediction is performed in context by the LLM; no Gaussian-process surrogate is fitted. Ties in the UCB score are broken pseudo-randomly.

Rubric versions. Versioning keeps the rubric that selects an experiment separate from a proposed dimension being tested on that experiment. Every decision is stored with its rubric version, judge and predictor settings, and available history. Adoption appends d, increments the rubric version, and merges the already measured values of d into the active profiles; any missing values are measured before they are used.

Table 3: Hyperparameters of RIGOR in all experiments.

Hyperparameter	Symbol	Value
Rubric authoring		
Independent drafts merged into the initial rubric		3
Dimensions of the initial rubric	K	15–20
Named options of a categorical dimension		3–8
Measurement		
Judge draws per candidate	J	3
Maximum calibration examples shown to the judge		3
Generation		
Candidates per slate	M	6
Maximum single-change children per slate	M_{c}	3
Recent designs shown in full		3
Prediction and selection		
Predictor draws	S	3
Nominal coverage of the prediction interval	$1-\alpha$	80%
Initial exploration weight	β_0	1.5
Prospective gate		
Outcomes before the first reflection	$t_{\min}$	4
Outcomes per window	W	5
Required relative MAE reduction	ρ	5%
Maximum adopted dimensions per run	$A_{\max}$	5
Budget		
Evaluations per run	T	24

Table 4: Evaluation tasks. All objectives are maximized. Scientific-design tasks use pretrained evaluators, the same for all methods; MLS-Bench tasks use the official Verifiers.

Task	Candidate design	Evaluation objective
Scientific design		
Nanohelix	Pitch, fiber radius, turns, helix radius	Predicted chiroptical g-factor
ChEMBL	Molecular SMILES string	Predicted binding affinity (pChEMBL)
SuperCon	Chemical composition	Predicted critical temperature T_c (K)
TMC	Pd complex with four ligands	Predicted polarizability
MLS-Bench		
Clustering	Clustering algorithm	Combined score over 3 settings
Online bandit	Arm selection and update policy	Regret-derived score over 3 settings
Scaling law	Fitted loss-prediction model	Prediction quality over 3 scaling-law families
MOEA	Multi-objective evolutionary strategy	HV, IGD, and spread score over 4 problems

C TASK PROTOCOLS AND IMPLEMENTATION

Hyperparameters. Table 3 lists all algorithmic hyperparameters, which are fixed across all eight tasks and both foundation models. Each run starts from an empty history and a newly authored rubric of $K = 15$ to 20 dimensions. With $T = 24$, $t_{\min} = 4$, and $W = 5$, at most three prospective windows can complete in a run, so $A_{\max} = 5$ is never binding.

Tasks. Table 4 summarizes the candidate design and objective of each task.

Scientific-design tasks. A Nanohelix design specifies pitch, fiber radius, number of turns, and helix radius, constrained respectively to $[39.2, 208.0]$ nm, $[7.0, 56.5]$ nm, $[1, 8]$, and $[6.6, 92.4]$ nm. A ChEMBL design is a valid molecular SMILES string, and the evaluator returns its predicted binding affinity; admissible molecules have molecular weight in $[250, 500]$ Da, quantitative estimate of drug-likeness (QED) at least 0.5, $\log P \leq 5$, at most five hydrogen-bond donors, and at most ten acceptors. A SuperCon design is a material formula, and the evaluator returns its predicted critical temperature; a formula is admissible if all 45 of the evaluator's composition features satisfy $|z| \leq 1.4$ with respect to its training distribution. A TMC design is a Pd(II) complex with two neutral and two anionic ligands from a fixed ligand list, and the evaluator returns its predicted polarizability. A submitted design outside the feasible domain is not evaluated; it uses its unit and receives an outcome of zero.

MLS-Bench tasks. Each task asks for an algorithm called by the benchmark's fixed evaluation code under the official settings. The clustering algorithm receives standardized features and the number of clusters. An online-bandit design specifies the policy's initialization, arm selection, and update rule. A scaling-law design fits and applies a loss-prediction model for vocabulary scaling, learning-rate and batch-size scaling, and data-constrained scaling. An MOEA design specifies parent selection, variation, survival, and the generation loop. It is evaluated on ZDT1, ZDT3, DTLZ2, and DTLZ1 with population sizes 100, 100, 120, and 120 and generation counts 200, 200, 250, and 400, respectively; its score combines hypervolume (HV), inverted generational distance (IGD), and spread. The three runs of each setting are independent. Every evaluation runs with 4 CPU cores and 8 GB of memory; MOEA Verifier runs are limited to 720 s. These settings are identical across methods.

Execution and experimental records. The Executor implements the selected design within the task's editable interface. On MLS-Bench, a separate Verifier executes the official settings, checks that the fixed parts of the program are unchanged, and computes the scalar outcome. A fidelity audit compares the implementation with its declared design and, for a child, with its parent. The Executor can repair implementation errors and deviations found by the audit within the same submission. Each successful evaluation is stored with its final code, outcome, parent, audit decisions, and pre-execution forecasts, allowing every prospective decision to be reconstructed.

Shared task context. For code tasks, the shared task context contains the research question, the editable interface, and the source files of the initial program and its fixed calling code. The same context is available during rubric authoring, generation, measurement, prediction, and execution. Hidden benchmark targets and scoring data remain confined to the Verifier.

Baselines. PiEvo uses its official implementation with its default settings and its Qwen3-Embedding model. Our ReElicit implementation applies the method to designs written in text, uses the same Executor as RIGOR, and spends the budget in six batches of four evaluations, the first of which is the initial batch. Both baselines use the same foundation models, task descriptions, evaluators, feasibility constraints, and budget as RIGOR. All runs used Claude opus-4-7 and OpenAI gpt-5.5.

Budget accounting. One unit of budget is one complete experiment: a design is submitted once to the evaluator, and its outcome is recorded. On the scientific tasks, an experiment is one run of the pretrained evaluator. On MLS-Bench, it is one complete Verifier measurement of an implemented design. Every method receives 24 units in every run. The same rule applies to every method: a submitted design that is invalid (it cannot be implemented in its task, or it violates the feasibility constraints) wastes its unit and receives an outcome of zero. During the experiments, we observe that every one of the 1,152 RIGOR designs is valid and receives a score, so no RIGOR outcome is zero. The rubric-based representation keeps generated designs valid: its dimensions name concrete parameters and mechanisms that the judge must locate in the design text, the hypothesis agent writes every candidate in terms of these dimensions and sees the measured profiles of executed designs, and a single-change child starts from the implementation of an evaluated parent. Among the 2,304 baseline outcomes, 33 are zero (1.4%): 27 invalid designs (23 of ReElicit and 4 of PiEvo) and 6 valid designs that the Verifier scored zero (3 of ReElicit and 3 of PiEvo). Best-so-far curves aggregate the cumulative maximum of each run, and current-step curves aggregate the outcome at each step.

D TRAJECTORY STATISTICS

Table 5 reports the trajectory statistics underlying Section 4.3. The mean incumbent is $A = 24^{-1} \sum_{t=1}^{24} \max_{i \le t} y_i$, and the mean evaluated outcome is $Q = 24^{-1} \sum_{t=1}^{24} y_t$. The late mean Q_{late} and the late within-run standard deviation S_{late} use evaluations 13–24. We compute S_{late} as the population standard deviation of a run's twelve late outcomes, whereas Table 1 reports the sample standard deviation of the three final best scores. All comparisons are within a task and foundation model.

D.1 ENLARGED SEARCH TRAJECTORIES

For readability, Figures 6–9 reproduce the trajectory plots from the main text at larger scale.

Table 5: Trajectory statistics for all 16 settings, averaged over three runs. The layout follows Table 1: boldface marks the best mean in each setting, and the RIGOR columns are shaded.

	Opus 4.7			GPT-5.5		
Task	RIGOR	PiEvo	ReElicit	RIGOR	PiEvo	ReElicit
Mean incumbent A						
Scientific design						
Nanohelix	**1.077**	0.949	0.991	**1.055**	1.048	0.941
ChEMBL	**4.862**	4.647	4.630	4.694	4.612	**4.736**
SuperCon	108.7	**111.0**	109.4	110.4	**112.7**	109.6
TMC	**443.3**	401.7	428.8	**464.1**	417.0	431.0
MLS-Bench						
Clustering	**0.567**	0.516	0.554	0.542	0.530	**0.560**
Online bandit	0.730	0.452	**0.731**	**0.957**	0.844	0.831
Scaling law	**0.503**	0.500	0.474	**0.571**	0.553	0.553
MOEA	**0.462**	0.385	0.388	**0.488**	0.382	0.420
Mean evaluated outcome Q						
Scientific design						
Nanohelix	**1.067**	0.848	0.855	**1.053**	0.923	0.906
ChEMBL	**4.616**	4.006	4.460	**4.592**	4.050	4.046
SuperCon	**106.3**	96.2	103.7	**109.4**	99.5	99.2
TMC	**424.0**	348.7	388.3	**451.9**	326.5	387.4
MLS-Bench						
Clustering	**0.534**	0.286	0.457	**0.509**	0.271	0.468
Online bandit	**0.697**	0.302	0.476	**0.943**	0.519	0.659
Scaling law	**0.498**	0.379	0.410	**0.561**	0.452	0.365
MOEA	**0.428**	0.194	0.307	**0.449**	0.225	0.309
Late mean outcome Q_{late}						
Scientific design						
Nanohelix	**1.096**	0.924	0.883	**1.071**	0.923	0.921
ChEMBL	**4.678**	4.231	4.621	**4.705**	4.120	4.499
SuperCon	**109.0**	101.9	107.0	**110.8**	106.3	103.1
TMC	**449.2**	369.3	406.1	**459.7**	344.6	394.4
MLS-Bench						
Clustering	**0.552**	0.257	0.490	**0.542**	0.257	0.504
Online bandit	**0.757**	0.444	0.535	**0.948**	0.593	0.719
Scaling law	**0.537**	0.389	0.434	**0.593**	0.457	0.366
MOEA	**0.455**	0.234	0.343	**0.482**	0.303	0.358
Late within-run standard deviation S_{late} (lower is better)						
Scientific design						
Nanohelix	**0.010**	0.113	0.162	**0.009**	0.263	0.026
ChEMBL	0.267	0.378	**0.220**	**0.163**	0.554	0.528
SuperCon	**0.9**	16.3	4.9	**0.6**	8.2	12.5
TMC	**18.0**	48.4	47.9	**16.2**	84.3	34.6
MLS-Bench						
Clustering	**0.035**	0.184	0.063	**0.023**	0.149	0.068
Online bandit	**0.038**	0.276	0.122	**0.075**	0.312	0.210
Scaling law	**0.012**	0.122	0.082	**0.016**	0.064	0.237
MOEA	**0.035**	0.157	0.090	**0.023**	0.170	0.102

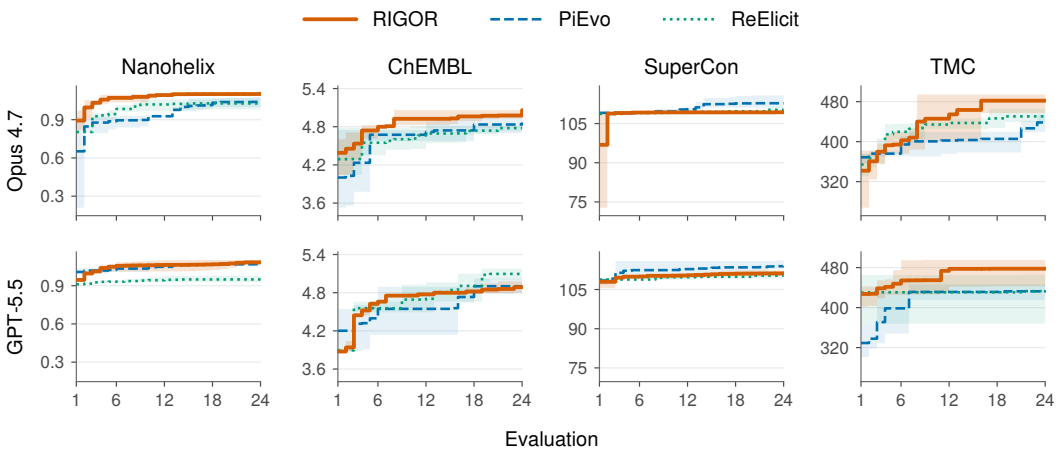

Figure 6: **Best-so-far score on the scientific-design tasks.** Enlarged version of Figure 3 (left). Lines show the mean over three runs; shaded regions span the minimum and maximum.

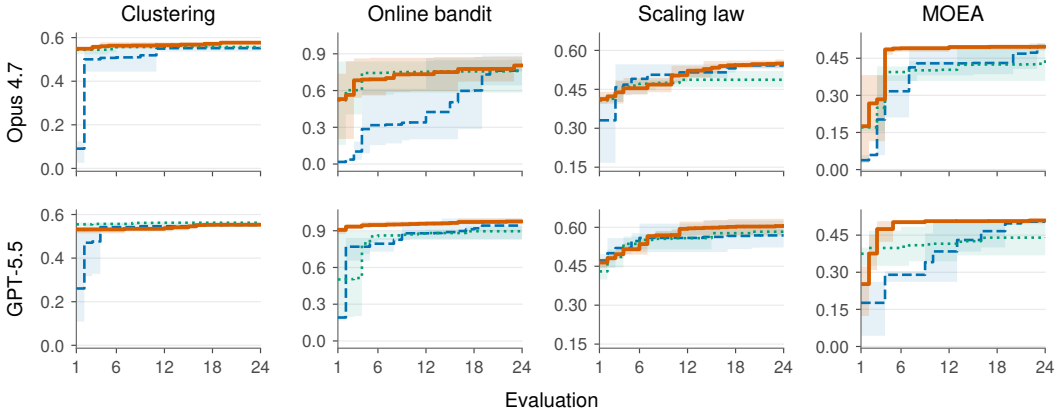

Figure 7: **Best-so-far score on the MLS-Bench tasks.** Enlarged version of Figure 3 (right). Lines show the mean over three runs; shaded regions span the minimum and maximum. Line styles follow Figure 6.

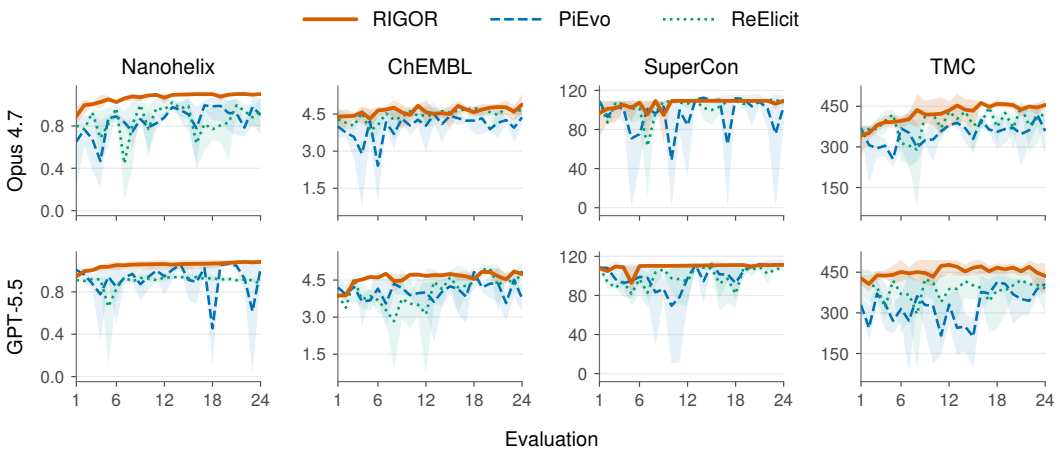

Figure 8: **Outcome at each step on the scientific-design tasks.** Enlarged version of Figure 4 (left). Lines show the mean over three runs; shaded regions span the minimum and maximum.

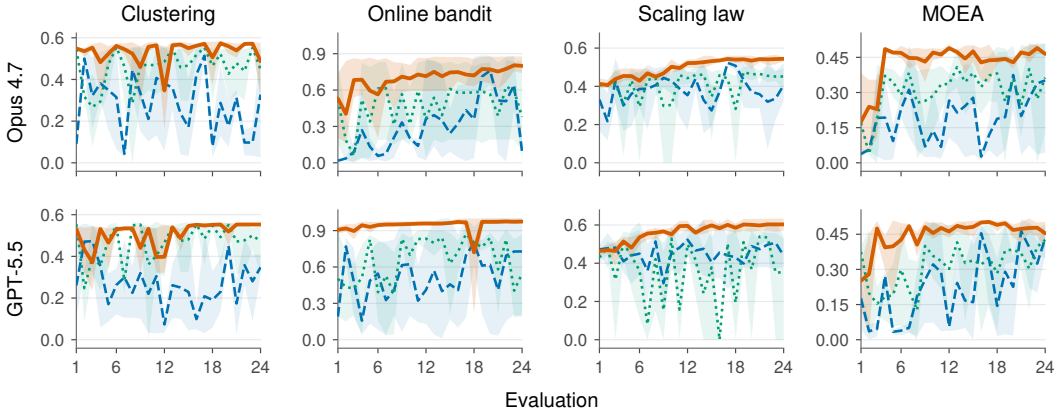

Figure 9: **Outcome at each step on the MLS-Bench tasks.** Enlarged version of Figure 4 (right). Lines show the mean over three runs; shaded regions span the minimum and maximum. Line styles follow Figure 8.

Table 6: Gap to the best method: relative difference, in percent, between a method's mean final best score and the highest mean in the setting. A value of 0.0 marks the leading method. The last two rows report the mean and largest gap over the eight tasks; boldface marks the smallest value.

	Opus 4.7			GPT-5.5		
Task	RIGOR	PiEvo	ReElicit	RIGOR	PiEvo	ReElicit
Scientific design						
Nanohelix	**0.0**	5.6	6.7	**0.0**	1.3	12.4
ChEMBL	**0.0**	3.8	5.6	4.5	4.2	**0.0**
SuperCon	3.1	**0.0**	2.3	2.4	**0.0**	3.2
TMC	**0.0**	9.0	6.5	**0.0**	9.1	9.4
MLS-Bench						
Clustering	**0.0**	4.3	3.4	1.6	1.8	**0.0**
Online bandit	**0.0**	5.3	5.7	**0.0**	3.4	8.1
Scaling law	**0.0**	1.4	11.4	**0.0**	6.0	3.7
MOEA	0.8	**0.0**	11.7	**0.0**	1.0	13.4
Mean over tasks	**0.5**	3.7	6.7	**1.1**	3.3	6.3
Largest	**3.1**	9.0	11.7	**4.5**	9.1	13.4

E Comparison with the best method in every setting

Table 6 reports the relative gap between each method's mean final best score and the highest mean in each setting. RIGOR has the highest mean in 11 of 16 settings and is within 4.5% of the leader in the remaining five. Across all settings, its average gap is 0.8%, compared with 3.5% for PiEvo and 6.5% for ReElicit; its largest gap is 4.5%, compared with 9.1% and 13.4%. RIGOR ranks first or second in 14 settings, and its mean rank is 1.44, compared with 2.06 for PiEvo and 2.50 for ReElicit. Its mean final best score exceeds that of PiEvo in 12 settings and that of ReElicit in 13. Where RIGOR leads, its margin over the stronger baseline ranges from 1.0% to 10.0%, with a mean of 4.3%. Where another method leads, RIGOR is within 0.8% to 4.5% of it, with a mean gap of 2.5%.

The five settings with another leading method show the same strengths as the rest of the evaluation. No baseline leads consistently: PiEvo leads three of these settings and ReElicit two, and these baselines trail the best method by up to 9.1% and 13.4% in other settings. In all five settings, RIGOR has the highest mean evaluated outcome and the lowest late within-run variation (Table 5), and the standard deviation of its final best score across runs is lower than that of the leading method (Table 1). RIGOR therefore delivers the best or nearly the best design in every setting, and it does so more reproducibly than the method that leads.

E.1 Consistency across tasks and models

Results across foundation models. All runs use the same configuration. RIGOR has the highest mean evaluated outcome in all eight Opus 4.7 settings and all eight GPT-5.5 settings, and the highest mean final best score in six and five settings, respectively. RIGOR therefore transfers between foundation models without retuning.

Reliable refinement within a design family. On SuperCon, RIGOR concentrates its evaluations within one productive design family. With Opus 4.7, all three runs identify the Hg–Ba–Ca–Cu–O cuprate family and refine compositions within it; their final best scores range from 109.28 to 109.38 K. Its late within-run standard deviation is 0.9 K, compared with 16.3 K for PiEvo and 4.9 K for ReElicit, and its mean evaluated outcome is 106.3 K, compared with 96.2 K and 103.7 K. With GPT-5.5, the corresponding mean evaluated outcomes are 109.4 K, 99.5 K, and 99.2 K, and the late within-run standard deviations are 0.6 K, 8.2 K, and 12.5 K. In a laboratory, where every composition must be synthesized and measured, this consistency means far fewer low-value experiments.

Table 7: Outcome predictor compared with forecasts using only earlier outcomes from the same run. Relative MAE below one favors the predictor.

Reference forecast	Relative MAE of the predictor	Runs with lower MAE
Mean of the earlier outcomes	0.456	47/48
Median of the earlier outcomes	0.621	46/48
Mean of the last three outcomes	0.698	45/48
Last outcome	0.725	43/48

Table 8: Selection evidence from the six RIGOR runs of each task. Relative MAE divides predictor MAE by the MAE of the historical-median reference (mean over runs, with sample standard deviation in gray); values below one favor the predictor.

Task	Relative MAE of the predictor	Runs with lower MAE	Exploratory UCB selections	Single-change children
Scientific design				
Nanohelix	0.344 ± 0.057	6/6	25/144	88/144
ChEMBL	0.567 ± 0.110	6/6	7/144	74/144
SuperCon	1.044 ± 0.860	5/6	35/144	75/144
TMC	0.441 ± 0.110	6/6	13/144	88/144
MLS-Bench				
Clustering	0.856 ± 0.096	6/6	58/144	99/144
Online bandit	0.555 ± 0.230	6/6	36/144	76/144
Scaling law	0.368 ± 0.120	6/6	48/144	84/144
MOEA	0.793 ± 0.188	5/6	26/144	102/144
All tasks	0.621	46/48	248/1,152	686/1,152

Strong late evaluations in algorithm and molecular design. On clustering with GPT-5.5, RIGOR has a late mean outcome of 0.542, compared with 0.257 for PiEvo and 0.504 for ReElicit, with late within-run standard deviations of 0.023, 0.149, and 0.068. On MOEA with Opus 4.7, its mean incumbent is 0.462, compared with 0.385 and 0.388, and its late mean outcome is 0.455, compared with 0.234 and 0.343. On ChEMBL with GPT-5.5, its late mean outcome is 4.705, compared with 4.120 and 4.499, with late within-run standard deviations of 0.163, 0.554, and 0.528.

F Forecast quality and robustness of the prospective gate

All analyses in this section use forecasts saved during the 48 RIGOR runs of the main comparison, with no additional model calls or experiments.

Reference forecasts. The outcome predictor is more accurate than each of four history-only reference forecasts in at least 43 of 48 runs (Table 7). As in Section 4.4, relative MAE is computed within each run over evaluations 2–24 before averaging over the 48 runs.

Selection evidence by task. Table 8 reports the selection analysis of Section 4.4 for each task.

Realized improvements. After the first evaluation of each run, the predictor forecasts an outcome above the incumbent for 374 of 1,104 selected candidates; 197 (52.7%) improve on the incumbent. Among the remaining 730 candidates, 178 (24.4%) improve.

Threshold and window robustness. Because every paired forecast was saved, we recompute gate decisions for alternative thresholds ρ and window lengths W (Tables 9 and 10). At $W = 5$, the gate rejects 60.5–94.2% of proposals as ρ varies from 0% to 20%; 11 of the 16 dimensions adopted at

Table 9: Gate decisions under alternative thresholds, recomputed from the saved forecast pairs of the 86 windows with $W = 5$. The setting used in the experiments is in bold.

Threshold ρ	0%	2.5%	**5%**	7.5%	10%	15%	20%
Dimensions adopted	34	22	**16**	12	11	7	5
Proposals rejected	60.5%	74.4%	**81.4%**	86.0%	87.2%	91.9%	94.2%

Table 10: Gate decisions using only the first W outcomes of each window, with $\rho = 5\%$. The setting used in the experiments is in bold.

Outcomes in the window W	1	2	3	4	**5**
Dimensions adopted	22	21	25	20	**16**
Same decision as with five outcomes	60/86	63/86	71/86	78/86	**86/86**
Adoptions that five outcomes reverse	16	14	12	6	**0**

Table 11: Prospective rubric refinement across the 48 RIGOR runs. Each window tests one frozen proposal on five subsequent outcomes within the same 24-evaluation budget; adoption requires at least a 5% relative MAE reduction.

Task	Completed windows	Dimensions adopted	Proposals rejected	MAE reduction of adopted dimensions
Scientific design				
Nanohelix	4	1	3	11.23%
ChEMBL	15	4	11	6.57–17.41%
SuperCon	8	2	6	6.71–40.38%
TMC	17	5	12	6.89–41.80%
MLS-Bench				
Clustering	11	0	11	N/A
Online bandit	12	2	10	14.25–31.29%
Scaling law	12	2	10	5.38–11.27%
MOEA	7	0	7	N/A
All tasks	86	16	70	5.38–41.80%

$\rho = 5\%$ also pass $\rho = 10\%$. Agreement with the five-outcome decision rises from 60/86 after one outcome to 78/86 after four. Of the 21 dimensions that a two-outcome window would adopt, the full five-outcome window rejects 14.

G Prospective decisions and traceable cases

Windows by task. Table 11 reports completed prospective windows, adopted dimensions, and rejected proposals for every task.

Adopted dimensions. Table 12 lists all dimensions adopted in the 48 RIGOR runs. The adopted dimensions include categorical mechanism identities and quantities computed from explicit design parameters.

Analysis details. Every selected design is matched to its pre-execution forecast, covering all 1,152 RIGOR evaluations. The historical-median reference is defined for evaluations 2–24 using no later outcomes. Relative MAE is computed within runs before aggregation, giving each run equal weight. A UCB selection counts as exploratory if its point forecast is strictly below the highest point forecast in its slate. The 430 paired gate forecasts produce the 16 adoptions and 70 rejections reported in Table 11.

Table 12: All 16 adopted dimensions. MAE reduction is $100(1 - E_1/E_0)$ on the fixed five-outcome window; the evaluation column gives the last evaluation of that window.

Task	Adopted dimension	Adopted after evaluation	MAE reduction
Opus 4.7			
ChEMBL	Heaviest non-fluorine halogen identity	16	17.41%
ChEMBL	Acid pharmacophore identity	21	15.82%
TMC	Isocyanide neutral ligand count	12	6.89%
TMC	Total oxygen-atom count across ligand SMILES	23	24.92%
Online bandit	Sliding-window bookkeeping scope	13	14.25%
GPT-5.5			
Nanohelix	Fractional turn end phase	20	11.23%
ChEMBL	Core scaffold family	9	6.57%
ChEMBL	Fluorine fraction among halogens	17	8.73%
SuperCon	Hg homologous cuprate member	9	6.71%
SuperCon	Bismuth share of reservoir-cation sublattice	15	40.38%
TMC	Duplicated anionic ligand identity	9	14.41%
TMC	Neutral ligand-pair identity	14	41.80%
TMC	Anionic ligand multiset identity	16	30.03%
Online bandit	Contextual forced initial coverage per arm	9	31.29%
Scaling law	Stage-1 bottleneck aggregation operator	11	11.27%
Scaling law	Stage-2 residual feature provenance and architecture	21	5.38%

Case 2: The full window determines what the rubric keeps TMC, GPT-5.5

Design trace: two saccharinate ligands fixed		*Outcome*
Eval. 4	Neutral ligands: collidine, collidine	456.5
Eval. 5	Child of 4: one collidine → PMe_2Ph	475.1
Eval. 6	Child of 5: second collidine → PMe_2Ph	**494.8**
Prospective gate		*MAE reduction*
Evals. 5–9: adopted	*Duplicated anionic ligand identity*	**14.41%**
Evals. 10–14: rejected	*Duplicated neutral ligand identity*	1.36%

The rejected proposal reduces cumulative MAE by 90.55% after two outcomes but only 1.36% over all five.

Figure 10: **A run traced through two prospective windows.** The gate adopts the first dimension (14.41% MAE reduction) and rejects the second, whose large improvement after two outcomes falls to 1.36% over five. Collidine denotes 2,4,6-trimethylpyridine and PMe_2Ph dimethylphenylphosphine.

G.1 Traceable cases

The full window determines what the rubric keeps. In the TMC run with GPT-5.5 traced in Figure 10, evaluations 4–6 keep the two saccharinate ligands fixed and replace the two collidine ligands with PMe_2Ph one at a time, increasing predicted polarizability from 456.5 to 494.8. A proposal for duplicated anionic-ligand identity reduces prospective MAE by 14.41% and becomes active after evaluation 9. A subsequent proposal for duplicated neutral-ligand identity reduces cumulative MAE by 90.55% after two outcomes but by only 1.36% after all five, so it is rejected. The gate judges each proposal on the outcomes of its own run: in another run of the same setting, a dimension for the unordered pair of neutral ligands reduces prospective MAE by 41.80% and is adopted (Table 12).