LEARNING WHAT TO INVESTIGATE NEXT: META-REASONING FOR LONG-HORIZON RESEARCH AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon research requires effective meta-reasoning, i.e., deciding what to investigate next as evidence accumulates. Learning this capability is challenging because the relevant decisions are embedded within long execution traces and their consequences are delayed. We introduce Meta-reasoning for Iterative Research Agents (**MIRA**), a hierarchical research harness whose outer loop curates context from a persistent Git-based research state and selects the objective for the next inner-loop execution agent. MIRA improves how models allocate additional inference: on IMOProofBench Advanced, it raises GPT-5.5's performance from 67.1% to 100% and helps it explore and pivot to new proof directions; in neural-architecture hill-climbing, it discovers improvements to both Residual Matrix and Loop Transformers. Isolating these decision boundaries also enables targeted credit assignment. We pretrain the language model as a generative critic that forecasts expected remaining return from partial research states across environments, then use the same model as both actor and its own critic in single-rollout asynchronous policy optimization over meta-reasoning decisions. We call this shared model **MIRA-AC**. By jointly training Qwen3.6-27B-Instruct to value the future potential of a research state and decide what to investigate next, while focusing policy-gradient updates on decisions comprising roughly 25% of generated output, MIRA-AC achieves gains of 10–15% across several autoresearch environments. It does so while training on readily available public hill-climbing signals, learning from its own research through proxy feedback rather than sparse or costly final-evaluation labels. Overall, MIRA-AC demonstrates that the strategy for conducting research can itself become a learned, reusable policy.

1 INTRODUCTION

Long-horizon research requires an agent to decide what to investigate next, which hypotheses to test, which results to pursue, when to change direction, and when to stop. These decisions depend on evidence accumulated over many experiments. That evidence can quickly exceed a model's context window, while the decisions themselves are embedded in long execution traces, making them difficult to isolate. Their consequences may emerge only after many subsequent experiments, further complicating credit assignment. An unsuccessful experiment may enable a subsequent discovery, while a promising direction may ultimately lead to a dead end. Effective research therefore requires not only executing investigations but also deciding how to allocate effort among them. This is a problem of *meta-reasoning*. Classical rational meta-reasoning asks which computation a resource-constrained agent should perform next and when it should stop (Russell & Wefald, 1991). We therefore ask: *Can we make the decisions that direct research explicit and improve them through experience?* Improving this allocation through experience requires three linked capabilities: (1) exposing the meta-reasoning decisions that steer a research trajectory (Sec. 2); (2) estimating the future potential of the partial research state at each decision (Sec. 3); and (3) assigning credit to those decisions so the policy itself can improve (Sec. 4). We study these requirements in turn.

Exposing research decisions with MIRA. MIRA is a hierarchical research harness that separates deciding what to investigate from executing the investigation. Each iteration, a meta-reasoning model curates evidence from a persistent Git record into an observation o_t and selects an action a_t: a work order specifying the next research objective, proposed approach, and success criteria, or a decision to stop. A separate agent executes the work order and commits its results. Both contexts

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.

reset between iterations, while the research record persists. MIRA thus exposes research-allocation decisions as explicit actions without requiring complete execution history in the active context.

Without additional policy training, MIRA solves all 30 IMOProofBench Advanced problems (Luong et al., 2025) in each of three GPT-5.5 trials, raising performance from 67.1% to 100% under the same runtime budget. Token-matched continuation recovers most of this aggregate gap, but on the remaining difficult problems MIRA more often explores alternative proof directions rather than repeatedly refining unsuccessful approaches. We also introduce **HillClimbBench** for open-ended neural-architecture research. For Residual Matrix Transformers (Mak & Flanigan, 2025), MIRA discovers learned per-cell gates that control how strongly each layer writes to different locations in the matrix-valued residual state. For Loop Transformers (Geiping et al., 2025), it combines a signed attention correction that can reinforce or suppress information with a direct path that retains the previous recurrent state, allowing each recurrent iteration to revise its state without overwriting it wholesale. Both designs improve upon their reference implementations.

Learning the value of partial research states. Explicit decisions provide a natural boundary for learning, but an investigation's immediate outcome may not reflect its eventual contribution. We therefore train a decision-level generative critic to estimate the expected remaining return under the continuation policy and research budget. From the same curated observation available to the policy, it uses the language model's native next-token distribution to represent beliefs over future returns. Decision-level prediction reduces held-out RMSE by 0.110 relative to its token-level generative counterpart. We also show that pretraining the critic across several environments provides a stronger prior that improves both zero-shot prediction and adaptation in unseen environments. After five target-environment episodes, this prior reduces RMSE by a further 0.053 compared with target-only training.

Training research decisions with MIRA-AC. We introduce **MIRA-AC**, a unified generative actor–critic initialized from a pretrained critic checkpoint. It jointly learns to value the future potential of the current research state and to decide what to investigate next. MIRA-AC reconstructs complete research episodes across iterations. The critic supplies value forecasts for computing dense decision-level advantages, enabling targeted credit assignment at meta-reasoning boundaries while abstracting away the surrounding execution traces as part of the environment transition. Policy loss is applied only to work-order and stopping decisions, not to context curation or executor generations. We also introduce a single-rollout asynchronous policy-optimization variant (Hou et al., 2026) with decoupled buffers for actor and critic training, allowing policy and value updates to proceed independently and propagate to ongoing research runs at subsequent meta-reasoning boundaries.

Improving autoresearch through reinforcement learning. We evaluate policy learning on symbolic regression, physics discovery, Bayesian causal discovery, and CPU-based neural-architecture research. By focusing policy-gradient updates on meta-reasoning decisions within MIRA's outer loop—which accounts for roughly 25% of total generated output—targeted credit assignment with MIRA-AC achieves 10–15% performance gains across several environments. Jointly training the same model to forecast remaining progress and choose the next research objective generally outperforms keeping the actor and critic separate. Training on readily available public hill-climbing signals also improves private-evaluation performance across all four environments. A single policy trained across environments further improves BNLearn without BNLearn actor updates, providing evidence that learned research strategy can transfer across environments.

Overall, MIRA-AC shows that by isolating the meta-reasoning decisions that steer long-horizon research and learning to value the partial states on which they are based, we can assign credit where it matters, allowing the policy to effectively allocate inference by learning what to investigate next.

2 MIRA: Making Research Decisions Explicit

We first ask whether organizing research around explicit decisions improves inference-time performance without additional policy training. We introduce MIRA and evaluate it on mathematical theorem proving and open-ended neural-architecture research.

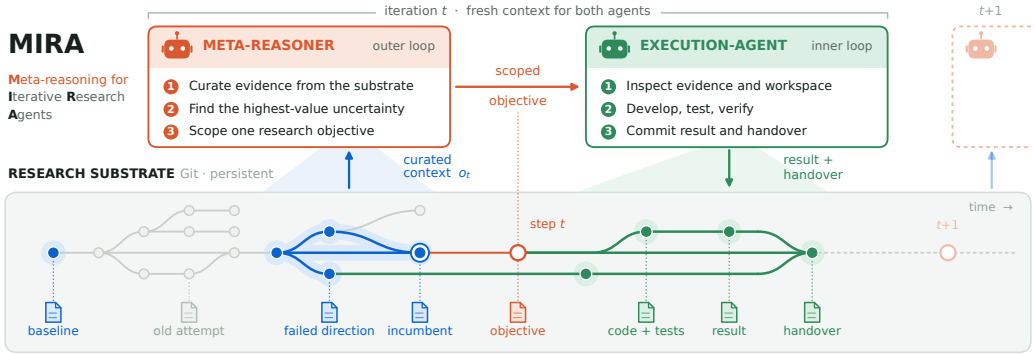


Figure 1: **MIRA separates research decisions from execution.** At each iteration, the outer loop curates evidence from a persistent Git record and selects one research objective. A fresh executor carries out the work order and commits its changes and handover. The updated record becomes the evidence source for the next decision.

2.1 A Research Decision as a Work Order

As shown in Figure 1, MIRA consists of an outer loop, an inner loop, and a persistent Git repository. The outer loop contains the meta-reasoning policy, which decides what to investigate next. The inner loop contains a fresh Codex executor (OpenAI, 2025) that performs the selected investigation. The Git repository serves as persistent external memory, storing the current solution, prior experiments and their results, and the artifacts needed to continue the research process.

At iteration t, the outer-loop policy retrieves relevant evidence from the repository and uses it to construct a curated observation o_t. Based on this observation, it selects an action a_t. The action issues a work order specifying the next research objective, proposed approach, and success criteria or, when submission is policy-controlled, stops the process and submits the current solution.

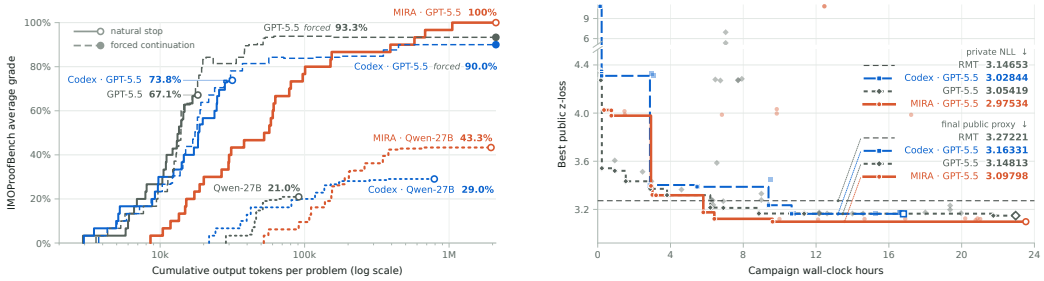
When the policy issues a work order, the inner loop launches a fresh Codex executor. The executor inspects the repository, modifies code or proofs, runs experiments, and evaluates candidates. This complete executor run constitutes one environment transition. The executor then commits its changes and writes a handover describing what it attempted, learned, and left unresolved.

The next outer-loop iteration constructs a new context from the updated repository. Both the outer-loop and inner-loop model contexts reset between iterations; only the persistent Git memory remains. The outer-loop procedure and inner-loop executor interface are shared across tasks, while task-specific repositories, tools, evaluation interfaces, and prompts are detailed in App. D.

Evaluation setup. We compare GPT-5.5, Codex, and MIRA. GPT-5.5 uses the task interface directly; Codex operates as one persistent tool-using session; and MIRA delegates each research objective to a fresh Codex inner loop. Codex and MIRA share the same executor, while MIRA adds explicit meta-reasoning and curated context resets. These comparisons evaluate the combined architecture, not its individual components. We test the three conditions on IMOProofBench (Luong et al., 2025) and HillClimbBench-RMT, and compare MIRA with persistent Codex on HillClimbBench-Loop. For theorem proving, we also include output-token-matched continuation controls.

2.2 Mathematical Theorem Proving

On IMOProofBench Advanced, we count a problem as solved only when the grader awards full credit. MIRA raises GPT-5.5's mean normalized grader score from 67.1% for the GPT-5.5 condition to 100%, solving all 30 problems in each of three trials. With Qwen3.6-27B-Instruct (Qwen Team, 2026), MIRA also more than doubles the model's score (Figure 2(a)). All conditions receive the same runtime budget, but MIRA makes better use of its inference budget to iterate on its solution, while the baselines submit much earlier. To determine whether its gains come from this allocation or simply from generating more tokens, we force the baselines to continue until they match MIRA's output-token count on each problem. This recovers most of the performance gap, but the residual



(a) IMOProofBench **(b)** HillClimbBench-RMT

Figure 2: **MIRA improves models' ability to conduct long-horizon theorem proving and neural-architecture hill climbing, outperforming comparable Codex and model-only baselines. (a)** Theorem-proving performance at natural stopping and under output-token-matched continuation. **(b)** Best loss observed during the matched 24-hour architecture-research campaigns.

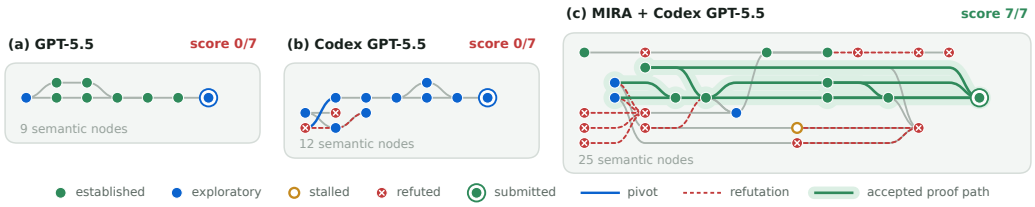


Figure 3: **MIRA improves the allocation of additional inference across proof directions.** On IMOProofBench Advanced Problem 23, GPT-5.5 and Codex pursue a small number of proof programs, while MIRA tests several directions, records refutations, and reaches the accepted proof.

gap is concentrated on four problems. On these problems, GPT-5.5 and Codex primarily extend or repair existing proof directions, whereas MIRA more often revisits earlier evidence and explores alternatives. We characterize this behavior using strategic inertia, the fraction of consecutive semantic steps within the same research direction. Across GPT-5.5 natural-stopping runs, mean inertia is 0.958 for GPT-5.5, 0.862 for Codex, and 0.772 for MIRA. MIRA's lower inertia means that it switches between proof directions more often. Figure 3 illustrates this behavior on Problem 23; App. E provides the metric definition and additional examples.

2.3 HillClimbBench: Neural-Architecture Research

We introduce HillClimbBench to evaluate MIRA in experimental research. Each environment starts from a task's paper-derived reference architecture and asks the agent to improve it through code changes and experiments. We study two open-ended architecture families; Sec. 4 uses a separate 50-task collection, HillClimbBench-CPU, for policy learning. Each task provides public test runs that the agent can use to hill-climb during research. These serve as a proxy for the final evaluation. After submission, a private evaluation produces the final design's private score on protected data.

2.3.1 Residual Matrix Transformer

The Residual Matrix Transformer (RMT) (Mak & Flanigan, 2025) replaces the usual vector-valued residual stream of the Transformer architecture (Vaswani et al., 2017) with a matrix that each layer reads from and writes to. The research problem is to regulate where attention and MLP updates write into this state without discarding its matrix structure. HillClimbBench-RMT asks the agent to improve the architecture under that constraint. We run matched 24-hour GPT-5.5 campaigns under the three orchestration conditions. GPT-5.5 stays close to conventional training and scaling choices, while Codex refines an RMT-specific normalization family. After more elaborate routing mechanisms prove ineffective at scale, MIRA shifts from normalization and cross-axis mixing to

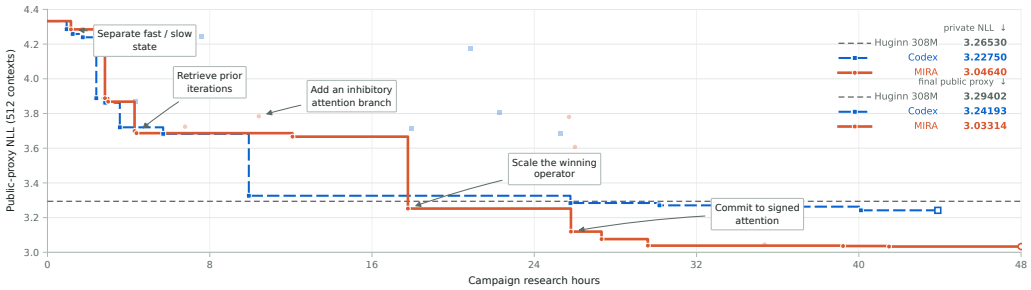


Figure 4: **MIRA enables GPT-6 Astra to improve a Loop Transformer through architectural hill climbing.** Across matched 48-hour campaigns, it outperforms persistent Codex and the reference implementation on both public-proxy and private evaluation, discovering a design that combines signed attention with direct retention of the prior recurrent state. Curves track the best public-proxy loss observed over time; faint markers show all evaluations, and hollow markers show where each run ends. Callouts identify major changes in MIRA's research direction. As in Figure 2, the two legend blocks separate each system's final public proxy from its private evaluation score.

local write control and selects learned per-cell residual-write gates. MIRA achieves the lowest public and private loss among the three campaigns (Figure 2(b)). App. F provides the matched controls, architecture, initialization, and ablations.

2.3.2 Loop Transformer

A Loop Transformer (Geiping et al., 2025) repeatedly applies a shared block, increasing effective depth without introducing distinct parameters at every iteration. Its central challenge is to revise the recurrent state without erasing information needed by later applications of the block. HillClimbBench-Loop asks the agent to improve this information flow. We compare matched 48-hour MIRA and persistent Codex campaigns using GPT-6 Astra. MIRA investigates fast and slow states, iteration history, and contrastive attention before selecting signed attention with recurrent-state retention. Signed attention contrasts two routing patterns over shared values, allowing information to be suppressed as well as reinforced, while the retention path preserves the prior recurrent state. This operator is related to Differential Attention (Ye et al., 2025), adapted to a weight-tied recurrent model. MIRA finishes ahead of persistent Codex on both the public proxy used during research and the separate private evaluation. Both campaigns also improve the public proxy over the task reference architecture. Figure 4 shows the progression of both research campaigns; App. G details the matched controls, operator, initialization, and selected designs.

These studies show how MIRA reallocates inference as evidence accumulates. By making each meta-reasoning decision explicit, MIRA also creates a natural credit-assignment boundary for learning from later outcomes how much progress remains possible.

3 Learning to Predict Remaining Research Progress

A decision's immediate outcome may not reflect its eventual contribution. We therefore ask whether expected remaining progress can be predicted at MIRA's decision boundaries and whether value learning transfers across research environments.

Returns and value. Let $T \leq H$ be the final decision in a research episode, where H is the maximum number of research decisions. The reward following decision t is r_t, and its undiscounted remaining return is $G_t = \sum_{u=t}^{T} r_u$. We consider two forms of feedback. Under private feedback, $Y \in \{0, 1\}$ is the binary outcome produced by private evaluation: the environment-specific solved indicator. For HillClimbBench-CPU, this means that the lower endpoint of the paired 95% confidence interval for improvement over the task reference architecture is positive. Reward is zero before termination and equals Y at the terminal step, so $G_t^{\text{priv}} = Y$. Under public experimental feedback, $q_t \in [0, 1]$ is the normalized public quality after decision t, and $\bar{q}_t$ is the best public quality observed through that

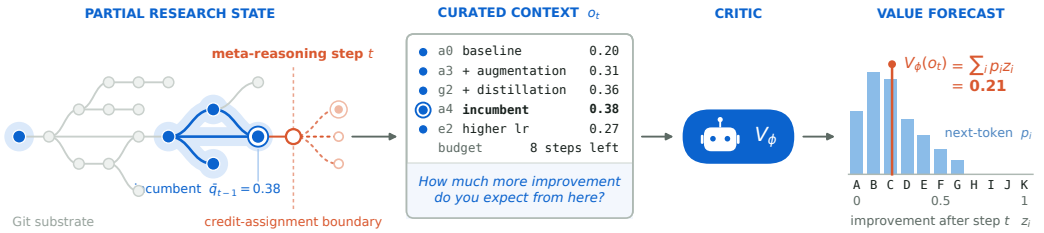


Figure 5: **Forecasting the future potential of a partial research state with a step-level generative critic.** The critic receives the curated observation o_t and predicts a distribution over the additional return expected from continuing the investigation. Its expectation is $V_\phi(o_t)$.

decision, including evaluations performed by its executor. The reward is the improvement over the previous incumbent, $r_t^{\text{pub}} = \bar{q}_t - \bar{q}_{t-1}$, and the remaining return is $G_t^{\text{pub}} = \bar{q}_T - \bar{q}_{t-1}$. This public metric provides a training signal but need not coincide with either the private score produced by final evaluation or the binary private outcome. For a fixed continuation policy and research procedure, the value of a curated observation is its expected remaining return, $V^\pi(o_t) = \mathbb{E}_\pi[G_t \mid o_t]$. Under private feedback, this is the probability of eventual success; under public feedback, it is the expected additional improvement over the current incumbent. Both depend on the continuation policy and remaining research budget.

3.1 A Generative Critic at Decision Boundaries

We train a step-level generative critic to estimate $V^\pi(o_t)$ from the observation available before the next work order. It predicts the expected return from continuing the investigation, not the maximum theoretically achievable improvement or the value of each candidate work order. We parameterize the critic using the language model's next-token distribution rather than an additional scalar head. Following the belief-elicitation procedure of Samanta et al. (2026b), we associate each support value z_i with a single-token label and interpret the normalized next-token probabilities over these labels as a predictive distribution over future returns:

$$p_\phi(z_i \mid o_t) = \frac{\exp \ell_{\phi,i}(o_t)}{\sum_{k=1}^{K} \exp \ell_{\phi,k}(o_t)}, \qquad V_\phi(o_t) = \sum_{i=1}^{K} p_\phi(z_i \mid o_t) z_i, \tag{1}$$

where $\ell_{\phi,i}(o_t)$ is the next-token logit for label i. We use a fixed discrete support for public return and a binary support for private outcomes. We distribute each continuous target between neighboring support points while preserving its expectation. The critic minimizes categorical KL loss (Bellemare et al., 2017). App. H specifies the projection. The critic receives the same curated observation as the policy and predicts value once per research decision, rather than at every generated token. We train it on Monte Carlo returns (Sutton & Barto, 2018) from complete research episodes. For critic-design and transfer experiments, we train on public experimental feedback, which is routinely available during research even when private evaluations are expensive or inaccessible.

3.2 A Prior for Partial Progress

We compare three step-level critics—scalar, categorical-head, and generative—with scalar and generative token-level baselines. For each environment, we form five task-disjoint splits of 50 support and 50 query episodes. Under matched conditions, each critic starts from the same Qwen-27B initialization and trains on all 50 support episodes. We evaluate it on the query episodes using mean per-episode RMSE. The step-level generative critic achieves the lowest held-out RMSE among the tested designs, reducing RMSE by 0.110 relative to its token-level generative counterpart (Figure 6(a)). We use this design in the subsequent transfer and policy-learning experiments.

We test whether critic pretraining improves prediction in an unseen research environment. For each target environment, we compare a target-only model with models initialized by pretraining on one source environment or on all remaining environments. The latter is leave-one-environment-out (LOO) pretraining. We then adapt each critic on 0, 1, 5, 10, 25, or 50 target-environment support

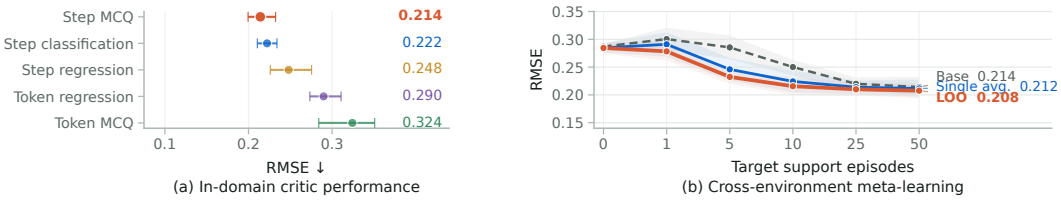


Figure 6: **Step-level generative prediction and cross-environment pretraining provide a stronger prior for partial progress. (a)** The step-level generative critic achieves the lowest held-out RMSE, and **(b)** LOO pretraining achieves the lowest RMSE at every target-support size, improving zero-shot prediction and adaptation to held-out environments. (95% paired-seed bootstrap CIs.)

episodes and evaluate it on the same 50 held-out query episodes. This measures how quickly each initialization improves as target-environment data increases. Across environments, LOO pretraining improves zero-shot prediction and adapts faster than either single-source pretraining or no pretraining. After adaptation on five target-environment episodes, it reduces RMSE by 0.053 compared with target-only training (Figure 6(b)). We use pretrained critics to assign delayed credit and train MIRA's research-decision policy.

4 MIRA-AC: Learning What to Investigate Next

We now ask whether value estimates at MIRA's decision boundaries can improve the policy that selects research objectives. We introduce MIRA-AC, a shared generative actor–critic that assigns delayed credit across complete research episodes and uses buffered asynchronous optimization to accommodate their variable durations.

4.1 A Unified Model for Actor and Critic

MIRA-AC uses one language model with shared parameters θ. In policy mode, it selects a work order or a stop decision from $\pi_\theta(\cdot \mid o_t)$; in critic mode, its next-token distribution yields $V_\theta(o_t)$, as defined in Sec. 3.1. We initialize the shared model from its pretrained target-environment critic checkpoint and update it using both policy and value learning objectives. Unlike a separate critic, whose predictions influence policy training through computed advantages, sharing parameters allows value learning from outcomes to update the same representation used for policy decisions.

4.2 Credit Assignment Across Research Decisions

Each MIRA iteration produces one transition (o_t, a_t, r_t, o_{t+1}). Context resets store successive transitions separately but do not terminate the research episode. We stitch successive iterations into complete episodes before computing returns and advantages. The critic learns from the undiscounted Monte Carlo return G_t. For policy training, we adapt GAE's λ parameter to the length of each reconstructed episode and compute advantages over the complete episode (Schulman et al., 2016). Because a live episode can span critic updates, adjacent decisions may use different versions of the critic. We correct for this mismatch before propagating advantages across version boundaries; App. J gives the exact construction. Stitching allows late outcomes to influence the advantages of earlier research decisions. Each $\hat{A}_t$ weights only the tokens of its corresponding work order or stop decision. Context curation, executor generations, and tool calls receive no direct policy loss.

4.3 Buffered Single-Rollout Asynchronous Policy Optimization

Research episodes vary in wall-clock duration. We collect them continuously, placing completed episodes in separate actor and critic buffers. Each buffer triggers updates to the same shared model independently, without waiting for synchronized rollout groups. A live episode switches to updated parameters only at its next meta-reasoning boundary and may therefore span several model versions. We record the actor behavior version, token log probabilities, and critic baselines at each decision. Following SAO (Hou et al., 2026), we use token-level importance ratios, mask policy-gradient up-

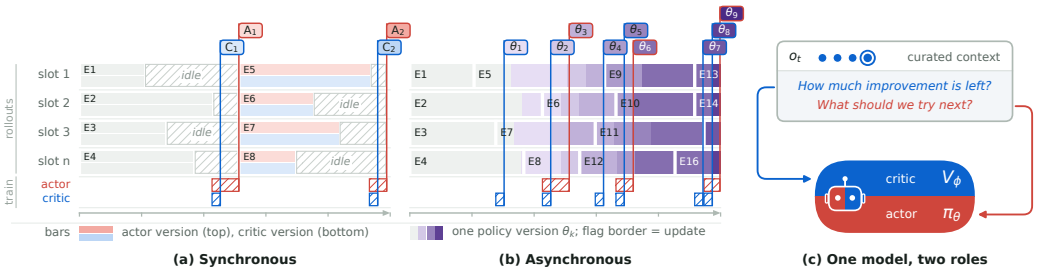


Figure 7: **Decoupled buffers allow MIRA-AC to learn asynchronously from variable-duration research episodes. (a)** Synchronous cohorts leave rollout slots idle while waiting for slow episodes. **(b)** MIRA-AC restarts freed slots immediately and updates the shared model whenever either the actor or critic buffer fills; live runs adopt new weights at the next meta-reasoning boundary. **(c)** The same language model serves as actor and critic, predicting remaining improvement and generating the next meta-reasoning step.

dates outside a specified ratio interval, and penalize behavior mismatch; we also average token losses within each decision. The complete objective and buffer details appear in App. I.

Experimental protocol. We evaluate on four environments: BNLearn causal discovery (Scutari, 2010), Physics Discovery (Wiemann et al., 2026), SRBench symbolic regression (Shojaee et al., 2025), and HillClimbBench-CPU. The first three each use 100 tasks. HillClimbBench-CPU has 50 paper-derived tasks, each with two independently initialized rollouts. Qwen-27B serves all roles, except that trained configurations use the learned policy at MIRA's meta-reasoning steps. We organize the comparison as an ablation ladder over supervision, credit granularity, critic parameterization, and actor–critic co-training. We begin with a separate actor–critic that uses token-level scalar regression and binary private outcomes. Holding the critic parameterization and granularity fixed, we replace terminal feedback with marginal improvements in the public metric. Under public-proxy supervision, i.e., training on public returns rather than the binary private outcome, we then move value prediction to MIRA's decision boundaries using step regression and replace scalar regression with the step-level generative critic. We evaluate the generative critic under both supervision regimes, first with separate policy and critic parameters and then with unified MIRA-AC training.

Token regression repeats the decision return target at each token prefix within the meta-reasoning action; step regression and the generative critic predict once per MIRA decision. In the separate-critic variants, the actor starts from the base policy and the critic from its pretrained checkpoint. MIRA-AC initializes the shared actor–critic from the pretrained critic checkpoint. Accordingly, this comparison evaluates the combined MIRA-AC design, not parameter sharing alone. All comparisons hold the MIRA harness, executor, tools, research horizon, inference budgets, and environment feedback fixed. Each run makes one pass over a fixed, ordered set of 100 training episodes, and we evaluate its final policy checkpoint. All reported results use private evaluation.

4.4 Experimental Results

Training with decision-level credit assignment improves long-horizon research. Across the ablation ladder, decision-level policy training improves final research outcomes over untrained MIRA in every environment. Under public-marginal supervision, the separate step-level generative critic also improves the primary success measure over token regression throughout the benchmark suite. App. Figure 13 shows the corresponding per-task hill climbs: under public-marginal supervision, MIRA-AC achieves a lower best-observed loss than untrained MIRA on 33 of 50 task families. Even including context curation, MIRA's outer-loop generations account for roughly 25% of model output, and policy-gradient loss is applied only to work-order and stopping decisions. Focusing learning on the decisions that allocate inference can therefore improve complete research trajectories without directly optimizing their much longer execution traces.

Jointly learning to forecast partial progress and act from the same research states produces a stronger policy. Ordinarily, the critic affects the policy only through the advantages used in the actor loss. In MIRA-AC, the shared critic loss also trains the policy's representation to recognize evidence

Table 1: **Autoresearch gains from training MIRA's meta-reasoning policy, measured by private evaluation.** Solved counts environment-specific successes; HillClimbBench-CPU reports paired wins over untrained MIRA. Private binary and Public marginal denote training feedback, not evaluation. Italics and bold mark the respective group bests; boxes mark the best overall counts. The final row trains one MIRA-AC policy across all environments except BNLearn, which is held out from actor training but included in critic pretraining.

		Symbolic Regression		**Physics Discovery**		**HillClimbBench-CPU**		**BNLearn**	
	Training variant	Solved	Score	Solved	Score	Wins	Score	Solved	Score
	MIRA (untrained)	50	0.9864	8	0.2118	—	0.7624	76	0.9672
Private binary	Token regression	55	0.9957	*9*	0.2327	*49*	*0.7594*	79	0.9567
	Step generative	58	*0.9993*	*9*	*0.2733*	43	0.7518	*82*	*0.9804*
	MIRA-AC	*65*	0.9879	6	0.1846	*49*	0.7481	56	0.9261
Public marginal	Token regression	55	0.9860	6	0.2463	49	0.7618	63	0.9002
	Step regression	50	0.9980	6	**0.3043**	46	0.7578	**84**	0.9733
	Step generative	59	0.9990	8	0.2463	53	0.7693	76	0.9106
	MIRA-AC	60	**0.9994**	**10**	0.2955	**58**	**0.8054**	48	0.9059
	Online actor training across all environments except BNLearn								
	MIRA-AC	**63**	0.9869	**10**	0.3028	55	0.7853	**84**	**0.9869**

predictive of future progress. Under public-marginal supervision, this dual objective improves over the separate generative critic in most environments. The critic loss therefore serves as an outcome-supervised auxiliary objective for learning what to investigate next.

Public progress supports reusable policy learning across environments. The BNLearn exception shows that environment-specific co-training can be brittle. A single MIRA-AC policy trained across the other environments improves over untrained MIRA throughout the benchmark suite, including on BNLearn, which receives no online actor updates. These improvements are measured by private evaluation, showing that public hill-climbing signals can train research behavior that carries across environments. Because BNLearn is included in critic pretraining, this provides evidence for transfer of actor learning across environments, not adaptation to an entirely unseen value domain.

Training changes how the agent conducts research. Across matched trajectories, an LLM judged MIRA-AC's decisions to contribute more to their respective final outcomes in 203 of 334 decisive comparisons (60.8%), indicating that training yielded more impactful meta-reasoning. Training increased the frequency with which the model reconsidered its approach, tested competing ideas, checked whether gains held, and stopped when further work was unlikely to help, while reducing repeated minor variations of an unproductive idea. See App. L for further discussion.

5 Conclusion and Limitations

Long-horizon autoresearch requires allocating inference as evidence accumulates, not merely carrying out investigations. MIRA exposes these choices as meta-reasoning decisions over curated partial states, separating what to investigate from execution. Through this boundary, MIRA-AC jointly trains one model to choose what to investigate and forecast progress, enabling decision-level credit assignment. Targeting policy updates to these decisions can improve long-horizon research even from imperfect public feedback without directly optimizing the much longer execution trajectories. Our method assumes the model can curate a sufficiently representative view of accumulated research; MIRA-AC trains objective selection, not curation, which remains an important direction for future work. We also hold the MIRA configuration and Codex-based inner-loop harness fixed, leaving cross-harness adaptation untested. Public hill-climbing signals may diverge from private evaluation. Finally, we train only LoRA adapters and have not tested asynchronous optimization on investigations lasting hours or days. Further limitations are in App. A, with related work in App. B. More broadly, MIRA-AC makes the strategy of conducting research itself an object of learning.

AI USAGE STATEMENT

This work studies how to train language-model agents to conduct research and, in some settings, improve other language models. The task-specific hypotheses, experiments, and results are therefore generated entirely by the agents being trained and evaluated. This does not extend to the paper's broader research, methods, claims, or conclusions, for which the authors take responsibility. Language models are also used as judges for qualitative analysis, including Physics Discovery explanation scoring and semantic exploration-graph annotation.

REFERENCES

Dhruv Agarwal, Bodhisattwa Prasad Majumder, Reece Adamson, Megha Chakravorty, Satvika Reddy Gavireddy, Aditya Parashar, Harshit Surana, Bhavana Dalvi Mishra, Andrew McCallum, Ashish Sabharwal, and Peter Clark. AutoDiscovery: Open-ended Scientific Discovery via Bayesian Surprise. In *Advances in Neural Information Processing Systems*, 2025. doi: 10.48550/arXiv.2507.00310. URL https://arxiv.org/abs/2507.00310. arXiv:2507.00310.

Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks, 2017. URL https://arxiv.org/abs/1703.00443.

Sercan O. Arik and Tomas Pfister. TabNet: Attentive interpretable tabular learning, 2019. URL https://arxiv.org/abs/1908.07442.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models, 2023. URL https://arxiv.org/abs/2312.04927.

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics, 2016. URL https://arxiv.org/abs/1612.00222.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 449–458. PMLR, 2017. URL https://arxiv.org/abs/1707.06887.

Jingyi Chai, Shuo Tang, Rui Ye, Yuwen Du, Xinyu Zhu, Mengcheng Zhou, Yanfeng Wang, Weinan E, Yuzhi Zhang, Linfeng Zhang, and Siheng Chen. SciMaster: Towards General-Purpose Scientific AI Agents, Part I. X-Master as Foundation: Can We Lead on Humanity's Last Exam?, July 2025. URL https://arxiv.org/abs/2507.05241. arXiv:2507.05241.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Lilian Weng, and Aleksander Mądry. MLE-bench: Evaluating Machine Learning Agents on Machine Learning Engineering. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6s5uXNWGIh. arXiv:2410.07095.

Jiefeng Chen, Bhavana Dalvi Mishra, Jaehyun Nam, Rui Meng, Tomas Pfister, and Jinsung Yoon. MARS: Modular Agent with Reflective Search for Automated AI Research. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. doi: 10.48550/arXiv.2602.02660. URL https://arxiv.org/abs/2602.02660. arXiv:2602.02660.

Minmin Chen. MinimalRNN: Toward more interpretable and trainable recurrent neural networks, 2017. URL https://arxiv.org/abs/1711.06788.

Minmin Chen, Jeffrey Pennington, and Samuel S. Schoenholz. Dynamical isometry and a mean field theory of RNNs: Gating enables signal propagation in recurrent neural networks, 2018a. URL https://arxiv.org/abs/1806.05394.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations, 2018b. URL https://arxiv.org/abs/1806.07366.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. ScienceAgentBench: Toward Rigorous Assessment of Language Agents for Data-Driven Scientific Discovery. In *International Conference on Learning Representations*, 2025. doi: 10.48550/arXiv.2410.05080. URL https://arxiv.org/abs/2410.05080. arXiv:2410.05080.

Ching-An Cheng, Andrey Kolobov, Dipendra Misra, Allen Nie, and Adith Swaminathan. LLF-Bench: Benchmark for Interactive Learning from Language Feedback, December 2023. URL https://arxiv.org/abs/2312.06853. arXiv:2312.06853.

Yizhe Chi, Deyao Hong, Dapeng Jiang, Tianwei Luo, Kaisen Yang, Boshi Zhang, Zhe Cao, Xiaoyan Fan, Bingxiang He, Han Hao, Weiyang Jin, Dianqiao Lei, Qingle Liu, Houde Qian, Bowen Wang, Situ Wang, Youjie Zheng, Yifan Zhou, Calvin Xiao, Eren Cai, and Qinhuai Na. Frontier-Eng: Benchmarking Self-Evolving Agents on Real-World Engineering Tasks with Generative Optimization, April 2026. URL https://arxiv.org/abs/2604.12290. arXiv:2604.12290.

Narsimha Chilkuri and Chris Eliasmith. Parallelizing legendre memory unit training, 2021. URL https://arxiv.org/abs/2102.11417.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2020. URL https://arxiv.org/abs/2009.14794.

Karl Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 2020–2027. PMLR, 2021. URL https://arxiv.org/abs/2009.04416.

Taco S. Cohen and Max Welling. Group equivariant convolutional networks, 2016. URL https://arxiv.org/abs/1602.07576.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers, 2018. URL https://arxiv.org/abs/1807.03819.

Simon Dennis, Rivaan Patil, Kevin Shabahang, and Hao Guo. Compiling Agentic Workflows into LLM Weights: Near-Frontier Quality at Two Orders of Magnitude Less Cost, May 2026. URL https://arxiv.org/abs/2605.22502. arXiv:2605.22502.

Yaxin Du, Xiyuan Yang, Zhifan Zhou, Wanxu Liu, Zixing Lei, Zimeng Chen, Fenyi Liu, Haotian Wu, Yuzhu Cai, Zexi Liu, Xinyu Zhu, WenHao Wang, Linfeng Zhang, Chen Qian, and Siheng Chen. DataMaster: Data-Centric Autonomous AI Research, May 2026. URL https://arxiv.org/abs/2605.10906. arXiv:2605.10906.

Yuanqi Du, Botao Yu, Tianyu Liu, Tony Shen, Junwu Chen, Jan G. Rittig, Kunyang Sun, Yikun Zhang, Aarti Krishnan, Yu Zhang, Daniel Rosen, Rosali Pirone, Zhangde Song, Bo Zhou, Cassandra Masschelein, Yingze Wang, Haorui Wang, Haojun Jia, Chao Zhang, Hongyu Zhao, Martin Ester, Nir Hacohen, Teresa Head-Gordon, Carla P. Gomes, Huan Sun, Chenru Duan, Philippe Schwaller, and Wengong Jin. Accelerating Scientific Discovery with Autonomous Goal-evolving Agents, December 2025. URL https://arxiv.org/abs/2512.21782. arXiv:2512.21782.

Shiyang Duan, Yuan Tian, Qi Bing, and Xiaowei Shao. Bayes-Entropy Collaborative Driven Agents for Research Hypotheses Generation and Optimization, August 2025. URL https://arxiv.org/abs/2508.01746. arXiv:2508.01746.

Shanghua Gao, Ada Fang, and Marinka Zitnik. AutoScientists: Self-Organizing Agent Teams for Long-Running Scientific Experimentation, May 2026. URL https://arxiv.org/abs/2605.28655. arXiv:2605.28655.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes, 2018. URL https://arxiv.org/abs/1807.01622.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *arXiv preprint arXiv:2502.05171*, 2025. URL https://arxiv.org/abs/2502.05171.

Shashwat Goel, Rishi Hazra, Dulhan Jayalath, Timon Willi, Parag Jain, William F. Shen, Ilias Leontiadis, Francesco Barbieri, Yoram Bachrach, Jonas Geiping, and Chenxi Whitehouse. Training AI Co-Scientists Using Rubric Rewards, December 2025. URL https://arxiv.org/abs/2512.23707. arXiv:2512.23707.

Alexander D. Goldie, Zilin Wang, Adrian Hayler, Deepak Nathani, Edan Toledo, Ken Thampiratwong, Aleksandra Kalisz, Michael Beukman, Hannah Erlebach, Alistair Letcher, Shashank Reddy, Clarisse Wibault, Theo Wolf, Charles O'Neill, Uljad Berdica, Nicholas Roberts, Saeed Rahmani, Roberta Raileanu, Shimon Whiteson, and Jakob N. Foerster. DiscoGen: Procedural Generation of Algorithm Discovery Tasks in Machine Learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. doi: 10.48550/arXiv.2603.17863. URL https://arxiv.org/abs/2603.17863. arXiv:2603.17863.

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 Tomasev, 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*, 2026. doi: 10.1038/s41586-026-10644-y. URL https://doi.org/10.1038/s41586-026-10644-y. arXiv:2502.18864.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines, 2014. URL https://arxiv.org/abs/1410.5401.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023. URL https://arxiv.org/abs/2312.00752.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. HiPPO: Recurrent memory with optimal polynomial projections, 2020. URL https://arxiv.org/abs/2008.07669.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces, 2021. URL https://arxiv.org/abs/2111.00396.

David Ha, Andrew Dai, and Quoc V. Le. HyperNetworks, 2016. URL https://arxiv.org/abs/1609.09106.

Kyle Helfrich, Devin Willmott, and Qiang Ye. Orthogonal recurrent neural networks with scaled Cayley transform, 2017. URL https://arxiv.org/abs/1707.09520.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Zhenyu Hou, Yujiang Li, Jie Tang, and Yuxiao Dong. Single-rollout asynchronous optimization for agentic reinforcement learning, 2026. URL https://arxiv.org/abs/2607.07508. arXiv:2607.07508.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2106.09685.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating Language Agents on Machine Learning Experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*. PMLR, 2024. doi: 10.48550/arXiv.2310.03302. URL https://proceedings.mlr.press/v235/huang24y.html. arXiv:2310.03302.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks, 2015. URL https://arxiv.org/abs/1506.02025.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1611.05397.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A general architecture for structured inputs & outputs, 2021. URL https://arxiv.org/abs/2107.14795.

Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. Tree Search for LLM Agent Reinforcement Learning. In *International Conference on Learning Representations*, 2026. doi: 10.48550/arXiv.2509.21240. URL https://arxiv.org/abs/2509.21240. arXiv:2509.21240.

Daniel R. Jiang, Ankur Samanta, Yukai Yang, Jalaj Bhandari, Rémi Munos, and Tyler Lu. Iterative GRPO: Batch-online multi-turn RL via single-turn RLHF, 2025a. URL https://arxiv.org/abs/2511.21638. arXiv:2511.21638.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. AIDE: AI-Driven Exploration in the Space of Code, February 2025b. URL https://arxiv.org/abs/2502.13138. arXiv:2502.13138.

Jiajie Jin, Yuyang Hu, Kai Qiu, Qi Dai, Chong Luo, Guanting Dong, Xiaoxi Li, Tong Zhao, Xiaolong Ma, Gongrui Zhang, Zhirong Wu, Bei Liu, Zhengyuan Yang, Linjie Li, Lijuan Wang, Hongjin Qian, Yutao Zhu, and Zhicheng Dou. Toward Generalist Autonomous Research via Hypothesis-Tree Refinement, June 2026. URL https://arxiv.org/abs/2606.11926. arXiv:2606.11926.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks, 2025. URL https://arxiv.org/abs/2510.04871.

Sanchit Kabra, Nikhil Abhyankar, Saaketh Desai, Prasad Iyer, and Chandan K. Reddy. LLM-AutoSciLab: Closed-Loop Scientific Discovery via Active Experimentation with LLMs, May 2026. URL https://arxiv.org/abs/2605.24043. arXiv:2605.24043 version: 1.

Nal Kalchbrenner, Ivo Danihelka, and Alex Graves. Grid long short-term memory, 2015. URL https://arxiv.org/abs/1507.01526.

Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. Independently recurrent neural network (IndRNN): Building a longer and deeper RNN, 2018. URL https://arxiv.org/abs/1803.04831.

Yu Li, Chenyang Shao, Xinyang Liu, Ruotong Zhao, Peijie Liu, Hongyuan Su, Zhibin Chen, Qinglong Yang, Anjie Xu, Yi Fang, Qingbin Zeng, Tianxing Li, Jingbo Xu, Fengli Xu, Yong Li, and Tie-Yan Liu. AutoSOTA: An End-to-End Automated Research System for State-of-the-Art AI Model Discovery, May 2026. URL https://arxiv.org/abs/2604.05550. arXiv:2604.05550.

Minhua Lin, Juncheng Wu, Zijun Wang, Zhan Shi, Yisi Sang, Bing He, Zewen Liu, Tianxin Wei, Zongyu Wu, Zhiwei Zhang, Dakuo Wang, Xiang Zhang, Benoit Dumoulin, Cihang Xie, Yuyin Zhou, Suhang Wang, and Hanqing Lu. Harness Updating Is Not Harness Benefit: Disentangling Evolution Capabilities in Self-Evolving LLM Agents, May 2026. URL https://arxiv.org/abs/2605.30621. arXiv:2605.30621.

Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the CoordConv solution, 2018. URL https://arxiv.org/abs/1807.03247.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y. Hou, and Max Tegmark. KAN: Kolmogorov-arnold networks, 2024. URL https://arxiv.org/abs/2404.19756.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention, 2020. URL https://arxiv.org/abs/2006.15055.

Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. PDE-Net: Learning PDEs from data, 2017. URL https://arxiv.org/abs/1710.09668.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery, September 2024. URL https://arxiv.org/abs/2408.06292. arXiv:2408.06292.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3:218–229, 2021. doi: 10.1038/s42256-021-00302-5. URL https://doi.org/10.1038/s42256-021-00302-5.

Thang Luong, Dawsen Hwang, Hoang H. Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, Alex Zhai, Clara Huiyi Hu, Henryk Michalewski, Jimin Kim, Jeonghyun Ahn, Junhwi Bae, Xingyou Song, Trieu H. Trinh, Quoc V. Le, and Junehyuk Jung. Towards robust mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025. doi: 10.18653/v1/2025.emnlp-main.1794. URL https://aclanthology.org/2025.emnlp-main.1794/.

Andreas Madsen and Alexander Rosenberg Johansen. Neural arithmetic units, 2020. URL https://arxiv.org/abs/2001.05016.

Brian Mak and Jeffrey Flanigan. Residual matrix transformers: Scaling the size of the residual stream. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2506.22696. arXiv:2506.22696.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with Gumbel-Sinkhorn networks, 2018. URL https://arxiv.org/abs/1802.08665.

Shambhavi Mishra, Gaurav Sahu, Marco Pedersoli, Laurent Charlin, Jose Dolz, and Christopher Pal. AInstein: Can LLMs Solve Research Problems From Parametric Memory Alone?, October 2025. URL https://arxiv.org/abs/2510.05432. arXiv:2510.05432.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pp. 1928–1937. PMLR, 2016. URL https://arxiv.org/abs/1602.01783.

Ryan L. Murphy, Balasubramaniam Srinivasan, Vinayak Rao, and Bruno Ribeiro. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs, 2018. URL https://arxiv.org/abs/1811.01900.

Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. Phased LSTM: Accelerating recurrent network training for long or event-based sequences, 2016. URL https://arxiv.org/abs/1610.09513.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. AlphaEvolve: A coding agent for scientific and algorithmic discovery, June 2025. URL https://arxiv.org/abs/2506.13131. arXiv:2506.13131.

OpenAI. Introducing Codex. https://openai.com/index/introducing-codex/, 2025.

Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences, 2023. URL https://arxiv.org/abs/2303.06349.

Alberto Pepe, Chien-Yu Lin, Despoina Magka, Bilge Acun, Yannan Nellie Wu, Anton Protopopov, Carole-Jean Wu, and Yoram Bachrach. Agentic Discovery of Neural Architectures: AIRA-Compose and AIRA-Design, May 2026. URL https://arxiv.org/abs/2605.15871. arXiv:2605.15871.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models, 2023. URL https://arxiv.org/abs/2302.10866.

Ori Press, Brandon Amos, Haoyu Zhao, Yikai Wu, Samuel K. Ainsworth, Dominik Krupke, Patrick Kidger, Touqir Sajed, Bartolomeo Stellato, Jisun Park, Nathanael Bosch, Eli Meril, Albert Steppi, Arman Zharmagambetov, Fangzhao Zhang, David Perez-Pineiro, Alberto Mercurio, Ni Zhan, Talor Abramovich, Kilian Lieret, Hanlin Zhang, Shirley Huang, Matthias Bethge, and Ofir Press. AlgoTune: Can Language Models Speed Up General-Purpose Numerical Programs?, October 2025. URL https://arxiv.org/abs/2507.15887. arXiv:2507.15887.

Rushi Qiang, Yuchen Zhuang, Yinghao Li, Dingu Sagar V. K, Rongzhi Zhang, Changhao Li, Ian Shu-Hei Wong, Sherry Yang, Percy Liang, Chao Zhang, and Bo Dai. MLE-Dojo: Interactive Environments for Empowering LLM Agents in Machine Learning Engineering, May 2025. URL https://arxiv.org/abs/2505.07782. arXiv:2505.07782.

Qwen Team. Qwen3.6-27B. Hugging Face model repository, 2026. URL https://huggingface.co/Qwen/Qwen3.6-27B/tree/6a9e13bd6fc8f0983b9b99948120bc37f49c13e9. Revision 6a9e13bd6fc8f0983b9b99948120bc37f49c13e9.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need, 2020. URL https://arxiv.org/abs/2008.02217.

Ben Rank, Hardik Bhatnagar, Ameya Prabhu, Shira Eisenberg, Karina Nguyen, Matthias Bethge, and Maksym Andriushchenko. PostTrainBench: Can LLM Agents Automate LLM Post-Training?, March 2026. URL https://arxiv.org/abs/2603.08640. arXiv:2603.08640.

David Rein, Joel Becker, Amy Deng, Seraphina Nix, Chris Canal, Daniel O'Connel, Pip Arnott, Ryan Bloom, Thomas Broadley, Katharyn Garcia, Brian Goodrich, Max Hasin, Sami Jawhar, Megan Kinniment, Thomas Kwa, Aron Lajko, Nate Rush, Lucas Jun Koba Sato, Sydney Von Arx, Ben West, Lawrence Chan, and Elizabeth Barnes. HCAST: Human-Calibrated Autonomy Software Tasks, March 2025. URL https://arxiv.org/abs/2503.17354. arXiv:2503.17354.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://doi.org/10.1038/s41586-023-06924-6.

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

Ankur Samanta, Akshayaa Magesh, Ayush Jain, Youliang Yu, Daniel Jiang, Kavosh Asadi, Kaveh Hassani, Paul Sajda, Jalaj Bhandari, and Yonathan Efroni. Credit assignment with resets in language model reasoning. *arXiv preprint arXiv:2605.25507*, 2026a. URL https://arxiv.org/abs/2605.25507. arXiv:2605.25507v2.

Ankur Samanta, Akshayaa Magesh, Tal Lancewicki, Ayush Jain, Youliang Yu, Paul Sajda, Kaveh Hassani, Aditya Modi, Daniel R. Jiang, and Yonathan Efroni. BayesBench: Evaluating LLM belief trajectories under multi-turn evidence accumulation, 2026b. URL https://arxiv.org/abs/2606.30850.

Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. $E(n)$ equivariant graph neural networks, 2021. URL https://arxiv.org/abs/2102.09844.

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, June 2025. URL https://arxiv.org/abs/2501.04227. arXiv:2501.04227.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1506.02438.

Marco Scutari. Learning bayesian networks with the bnlearn r package. *Journal of Statistical Software*, 35(3):1–22, 2010. doi: 10.18637/jss.v035.i03.

Zikang Shan, Han Zhong, Liwei Wang, and Li Zhao. Bringing value models back: Generative critics for value modeling in LLM reinforcement learning, 2026. URL https://arxiv.org/abs/2604.10701.

Parshin Shojaee, Ngoc-Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa D. Doan, and Chandan K. Reddy. LLM-SRBench: A new benchmark for scientific equation discovery with large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2504.10415. Oral presentation; arXiv:2504.10415.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions, 2020. URL https://arxiv.org/abs/2006.09661.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-Searcher: Incentivizing the Search Capability in LLMs via Reinforcement Learning, March 2025. URL https://arxiv.org/abs/2503.05592. arXiv:2503.05592.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI's Ability to Replicate AI Research, April 2025. URL https://arxiv.org/abs/2504.01848. arXiv:2504.01848.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018. URL https://mitpress.mit.edu/9780262039246/reinforcement-learning/.

Andrew Szot, Michael Kirchhof, Omar Attia, and Alexander Toshev. Expanding LLM Agent Boundaries with Strategy-Guided Exploration, March 2026. URL https://arxiv.org/abs/2603.02045. arXiv:2603.02045.

Corentin Tallec and Yann Ollivier. Can recurrent neural networks warp time?, 2018. URL https://arxiv.org/abs/1804.11188.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains, 2020. URL https://arxiv.org/abs/2006.10739.

Jos van der Westhuizen and Joan Lasenby. The unreasonable effectiveness of the forget gate, 2018. URL https://arxiv.org/abs/1804.04849.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2017. URL https://arxiv.org/abs/1710.10903.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks, 2015. URL https://arxiv.org/abs/1506.03134.

Aaron Voelker, Ivana Kajić, and Chris Eliasmith. Legendre memory units: Continuous-time representation in recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 32, pp. 15544–15553, 2019. URL https://papers.nips.cc/paper/9689-legendre-memory-units-continuous-time-representation-in-recurrent-neural-networks.

Chunhui Wan, Xunan Dai, Zhuo Wang, Minglei Li, Yanpeng Wang, Yinan Mao, Yu Lan, and Zhiwen Xiao. LoongFlow: Directed Evolutionary Search via a Cognitive Plan-Execute-Summarize Paradigm, December 2025a. URL https://arxiv.org/abs/2512.24077. arXiv:2512.24077.

Yi Wan, Jiuqi Wang, Liam Li, Jinsong Liu, Ruihao Zhu, and Zheqing Zhu. Rethinking the Design of Reinforcement Learning-Based Deep Research Agents, October 2025b. URL https://arxiv.org/abs/2510.15862. arXiv:2510.15862 version: 3.

Hanyu Wang, Yifan Lan, Bochuan Cao, Lu Lin, and Jinghui Chen. SkillGrad: Optimizing Agent Skills Like Gradient Descent, May 2026a. URL https://arxiv.org/abs/2605.27760. arXiv:2605.27760.

Miles Wang, Robi Lin, Kat Hu, Joy Jiao, Neil Chowdhury, Ethan Chang, and Tejal Patwardhan. FrontierScience: Evaluating AI's Ability to Perform Expert-Level Scientific Tasks, January 2026b. URL https://arxiv.org/abs/2601.21165. arXiv:2601.21165.

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. StepSearch: Igniting LLMs Search Ability via Step-Wise Proximal Policy Optimization, May 2025. URL https://arxiv.org/abs/2505.15107. arXiv:2505.15107.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks, 2014. URL https://arxiv.org/abs/1410.3916.

Matt L. Wiemann, Lindsay M. Smith, Peter Melchior, Siddharth Mishra-Sharma, Andrew Gordon Wilson, Pavel Izmailov, and Carolina Cuesta-Lázaro. DiscoverPhysics: Benchmarking LLMs for Out-of-the-Box Scientific Thinking, May 2026. URL https://arxiv.org/abs/2605.26087. arXiv:2605.26087.

Hjalmar Wijk, Tao Lin, Joel Becker, Sami Jawhar, Neev Parikh, Thomas Broadley, Lawrence Chan, Michael Chen, Josh Clymer, Jai Dhyani, Elena Ericheva, Katharyn Garcia, Brian Goodrich, Nikola Jurkovic, Holden Karnofsky, Megan Kinniment, Aron Lajko, Seraphina Nix, Lucas Sato, William Saunders, Maksym Taran, Ben West, and Elizabeth Barnes. RE-Bench: Evaluating frontier AI R&D capabilities of language model agents against human experts. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*. PMLR, 2025. doi: 10.48550/arXiv.2411.15114. URL https://proceedings.mlr.press/v267/wijk25a.html. arXiv:2411.15114.

Scott Wisdom, Thomas Powers, John R. Hershey, Jonathan Le Roux, and Les Atlas. Full-capacity unitary recurrent neural networks, 2016. URL https://arxiv.org/abs/1611.00035.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks?, 2018. URL https://arxiv.org/abs/1810.00826.

Wanqiao Xu, Allen Nie, Ruijie Zheng, Aditya Modi, Adith Swaminathan, and Ching-An Cheng. Formalizing Learning from Language Feedback with Provable Guarantees. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. doi: 10.48550/arXiv.2506.10341. URL https://arxiv.org/abs/2506.10341. arXiv:2506.10341.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist-v2: Workshop-Level Automated Scientific Discovery via Agentic Tree Search, April 2025. URL https://arxiv.org/abs/2504.08066. arXiv:2504.08066.

Xu Yang, Xiao Yang, Shikai Fang, Yifei Zhang, Jian Wang, Bowen Xian, Qizheng Li, Jingyuan Li, Minrui Xu, Yuante Li, Haoran Pan, Yuge Zhang, Weiqing Liu, Yelong Shen, Weizhu Chen, and Jiang Bian. R&D-Agent: An LLM-Agent Framework Towards Autonomous Data Science, October 2025. URL https://arxiv.org/abs/2505.14738. arXiv:2505.14738.

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, Kai Qiu, Yuqing Yang, Dongdong Chen, Xue Yang, and Chong Luo. SkillOpt: Executive Strategy for Self-Evolving Agent Skills, May 2026. URL https://arxiv.org/abs/2605.23904. arXiv:2605.23904.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.05258. Oral presentation.

Seungil You, David Ding, Kevin Canini, Jan Pfeifer, and Maya Gupta. Deep lattice networks and partial monotonic functions, 2017. URL https://arxiv.org/abs/1709.06680.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets, 2017. URL https://arxiv.org/abs/1703.06114.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2408.15240.

Bingchen Zhao, Despoina Magka, Minqi Jiang, Xian Li, Roberta Raileanu, Tatiana Shavrina, Jean-Christophe Gagnon-Audet, Kelvin Niu, Shagun Sodhani, Michael Shvartsman, Andrei Lupu, Alisia Lupidi, Edan Toledo, Karen Hambardzumyan, Martin Josifoski, Thomas Foster, Lucia Cipolina-Kun, Abhishek Charnalia, Derek Dunfield, Alexander H. Miller, Oisin Mac Aodha, Jakob Foerster, and Yoram Bachrach. The Automated LLM Speedrunning Benchmark: Reproducing NanoGPT Improvements, June 2025. URL https://arxiv.org/abs/2506.22419. arXiv:2506.22419.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language Agent Tree Search Unifies Reasoning Acting and Planning in Language Models. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. doi: 10.48550/arXiv.2310.04406. URL https://arxiv.org/abs/2310.04406. arXiv:2310.04406.

Julian Georg Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. Recurrent highway networks, 2016. URL https://arxiv.org/abs/1607.03474.

Qiran Zou, Hou Hei Lam, Wenhao Zhao, Tingting Chen, Yiming Tang, Samson Yu, Yingtao Zhu, Srinivas Anumasa, Zufeng Zhang, Tianyi Zhang, Chang Liu, Zhengyao Jiang, Anirudh Goyal, and Dianbo Liu. FML-bench: A Controlled Study of AI Research Agent Strategies from the Perspective of Search Dynamics, May 2026. URL https://arxiv.org/abs/2605.17373. arXiv:2605.17373.

APPENDIX CONTENTS

Context and limitations

A Limitations and Future Work . . . 21
B Related Work . . . 21

Experimental setup

C Experimental Setup . . . 23
D Additional MIRA Details . . . 26

Additional MIRA analyses

E Semantic Exploration Graphs and Strategic Inertia . . . 27
F RMT Architecture Identified by MIRA . . . 30
G Loop Transformer Search and Selected Designs . . . 31

Critic and policy-learning details

H Critic Details . . . 31
I Implementation Details . . . 33
J Trajectory Stitching Across Artificial Boundaries . . . 34
K Additional MIRA-AC Results . . . 35
L Behavioral Analysis of MIRA-AC . . . 35

A Limitations and Future Work

Although evidence curation and objective selection are both components of the meta-reasoning step, MIRA-AC leaves the curation policy fixed and trains only objective selection. Existing curation supports both hill-climbing and policy learning, but cannot learn from downstream outcomes which evidence proved decision-relevant. A natural next step is to train the complete meta-reasoning step, extending credit assignment to both the evidence surfaced and the research objective selected.

Our experiments use a single MIRA configuration with the same Codex-based inner-loop harness across research environments. This controls the research interface while varying the task, but we do not evaluate whether trained policies adapt across different execution scaffolds or evolving inner-loop harnesses. Training across heterogeneous and changing harnesses would test whether learned meta-reasoning remains useful as execution capabilities change.

Autoresearch requires frequent, inexpensive signals for hill-climbing, whereas private evaluations may be costly or available only at the end of an investigation. Their alignment is therefore an important part of environment design: a poor public proxy can direct research toward improvements that do not transfer to private evaluation. Nevertheless, our results show that imperfect public proxy rewards can improve private evaluation performance.

Natural next steps also include scaling MIRA-AC from LoRA (Hu et al., 2022) to full-parameter fine-tuning and testing buffered asynchronous optimization when individual investigations last hours or days, so feedback arrives slowly and live episodes span more policy versions. At that scale, actor and critic losses update the same parameters but need not produce aligned gradients, making their scheduling and stable co-optimization important questions. Future work should also measure how targeted optimization of meta-reasoning affects general reasoning beyond autoresearch.

B Related Work

Rational meta-reasoning. MIRA builds on the classical account in Principles of Metareasoning (Russell & Wefald, 1991), which asks which computation a resource-bounded agent should perform next and when it should stop. In long-horizon research, the relevant computations are investigations: testing a hypothesis, revising an implementation, gathering evidence, or terminating the search. MIRA makes these allocation decisions explicit as work orders, thereby turning a classical meta-reasoning problem into a trainable sequential decision process.

Benchmarks for autonomous research. Recent benchmarks expose complementary aspects of long-horizon research. MLAgentBench (Huang et al., 2024) and MLE-bench (Chan et al., 2025) evaluate iterative machine-learning engineering over executable tasks, while RE-Bench (Wijk et al., 2025) compares agents with human experts over increasing time budgets. PaperBench (Starace et al., 2025) evaluates end-to-end replication using author-developed hierarchical rubrics, and HCAST (Rein et al., 2025) grounds task difficulty in the time required by skilled humans. ScienceAgentBench (Chen et al., 2025) similarly derives data-driven discovery tasks from published scientific workflows.

Other environments broaden either the training interface or the space of research problems. MLE-Dojo (Qiang et al., 2025) provides interactive Kaggle environments with normalized rewards suitable for collecting learning trajectories; PostTrainBench (Rank et al., 2026) extends autoresearch to coupled post-training decisions; and FrontierScience (Wang et al., 2026b) targets expert-level scientific tasks. DiscoGen (Goldie et al., 2026) supplies procedurally generated algorithm-discovery tasks with meta-train/meta-test splits, whereas AlgoTune (Press et al., 2025) evaluates measurable speedups to established numerical programs. Finally, Frontier-Eng (Chi et al., 2026) characterizes how improvements diminish over extended engineering runs, and FML-bench (Zou et al., 2026) compares search strategies across controlled landscapes. In addition to introducing HillClimbBench, we cast several existing environments as instances of a common research decision process and ask whether research strategy can be learned across trajectories.

Scientific and algorithmic discovery agents. Several systems automate substantial portions of the research lifecycle. The AI Scientist (Lu et al., 2024) and AI Scientist-v2 (Yamada et al., 2025) combine experiment execution, evaluation, and paper generation, with the latter adding tree search

and structured memory. FunSearch (Romera-Paredes et al., 2024) established language-model-guided evolution as a mechanism for discovering programs, and AlphaEvolve (Novikov et al., 2025) scales this approach through model ensembles, evaluation cascades, and quality-diversity search. SAGA (Du et al., 2025) extends automation to the evolution of scientific objectives, while the AI co-scientist (Gottweis et al., 2026) and AutoDiscovery (Agarwal et al., 2025) organize hypothesis development through evolutionary selection and Bayesian surprise, respectively.

Research agents also differ in how they divide and retain work. R&D-Agent (Yang et al., 2025) separates researcher and developer roles, AutoScientists (Gao et al., 2026) coordinates decentralized teams through shared experimental memory, and MARS (Chen et al., 2026) performs budget-aware tree search over repository states. AutoSOTA (Li et al., 2026) targets reproduction and improvement of existing models, LLM-AutoSciLab (Kabra et al., 2026) casts discovery as active experimental design, and Arbor (Jin et al., 2026) refines explicit hypothesis trees. DiscoverPhysics (Wiemann et al., 2026) evaluates open-ended scientific reasoning in physics, whereas AInstein (Mishra et al., 2025) studies iterative critique using parametric knowledge alone. These systems primarily improve a fixed model through inference-time organization. MIRA is compatible with such execution mechanisms, but our central concern is learning the policy that chooses the next research objective from prior experience.

Search, memory, and evolving agent structure. Existing agents encode a wide range of search biases. AIDE (Jiang et al., 2025b) uses explicit drafting, improvement, and debugging operators; LATS (Zhou et al., 2024) combines tree search with acting, planning, reflection, and model-generated values; and LoongFlow (Wan et al., 2025a) couples plan–execute–summarize cycles with evolutionary archives. DataMaster (Du et al., 2026) schedules exploration over the available budget, AIRA (Pepe et al., 2026) applies UCT-style search to neural-architecture discovery, and SciMaster (Chai et al., 2025) distributes exploration across multiple agents. Automated AI-research Speedrunning (Zhao et al., 2025) provides a controlled autoresearch setting, while Agent Laboratory (Schmidgall et al., 2025) organizes research into fixed phases. The diversity of successful scaffolds, together with the strategy-dependent results of Frontier-Eng (Chi et al., 2026) and FML-bench (Zou et al., 2026), motivates selecting research actions conditionally rather than committing to one search heuristic for an entire investigation.

A complementary line of work changes the agent's persistent artifacts. SkillGrad (Wang et al., 2026a) treats trajectory diagnoses as textual gradients, and SkillOpt (Yang et al., 2026) uses bounded edits and validation to improve reusable skill documents. Harness self-evolution analysis (Lin et al., 2026) distinguishes the ability to modify a harness from the ability to benefit from those modifications, while Compiling Agentic Workflows into LLM Weights (Dennis et al., 2026) shows that procedural workflows can be distilled into model weights. MIRA keeps the execution harness fixed in our learning experiments and instead updates the model that allocates research effort; evolving the harness and learning the meta-reasoning policy are therefore complementary directions.

Feedback and value learning. Language feedback can convey substantially more structure than a scalar reward, as illustrated empirically by LLF-Bench (Cheng et al., 2023) and formalized by Formalizing Learning from Language Feedback with Provable Guarantees (Xu et al., 2026). Training AI Co-Scientists Using Rubric Rewards (Goel et al., 2025) uses rubric rewards to train research-plan generation, while HypoAgents (Duan et al., 2025) uses Bayesian belief and entropy updates to guide which hypotheses to refine. Our experiments instead use scalar, outcome-grounded targets: either a binary private outcome or changes in a public experimental metric. This choice isolates whether delayed research outcomes alone can train value estimates at MIRA's decision boundaries; versioned artifacts could support richer feedback in future work.

MIRA-AC also relates to shared actor–critic representation learning. A3C (Mnih et al., 2016) uses a shared backbone with separate policy and scalar-value outputs, while Phasic Policy Gradient (Cobbe et al., 2021) studies the balance between useful representation transfer and interference between policy and value objectives. UNREAL (Jaderberg et al., 2017) demonstrates how auxiliary prediction objectives can shape a shared policy representation. More recent generative approaches express verification or value estimation through language modeling itself: Generative Verifiers (Zhang et al., 2025) formulates verification as next-token prediction, and GenAC (Shan et al., 2026) uses a generative critic for value modeling. MIRA-AC similarly uses one language model and one next-token

interface for both research actions and return prediction, without adding a scalar head or maintaining a separately resident critic.

Credit assignment in multi-turn agents. Several methods localize outcome supervision within long interactions. Credit assignment can also proceed through a learned step-level critic: Iterative GRPO (Jiang et al., 2025a) fits a turn-level Q^π from logged Monte Carlo returns and uses it as the reward model for single-turn policy improvement in batch-online multi-turn settings. Credit Assignment with Resets (Samanta et al., 2026a) uses counterfactual continuations from intermediate states, and Tree-GRPO (Ji et al., 2026) obtains step-level relative signals from sibling branches that share a prefix. R1-Searcher (Song et al., 2025) learns retrieval behavior from outcome-only reinforcement learning, StepSearch (Wang et al., 2025) applies step-wise policy optimization to search-augmented reasoning, and strategy-guided exploration (Szot et al., 2026) operates over high-level language strategies. PokeeResearch (Wan et al., 2025b) studies reinforcement-learning design choices for deep-research agents. MIRA-AC differs in where it places the learning boundary: it reconstructs complete research episodes but applies policy loss only to the work-order and stopping decisions that allocate future computation. Its step-level generative critic assigns delayed credit at those boundaries, and buffered asynchronous optimization accommodates investigations with heterogeneous durations.

C Experimental Setup

We study four autoresearch environments: BNLearn causal discovery, Physics Discovery, SRBench symbolic regression, and HillClimbBench-CPU architecture research. In the reported policy evaluation, BNLearn, Physics, and SRBench each contain 100 tasks with one rollout per task. HillClimbBench-CPU contains 50 distinct paper-derived tasks with two independently initialized rollouts per task. Each rollout permits at most 20 MIRA iterations.

C.1 Shared interaction protocol

Each task is an isolated, persistent workspace. At every iteration, a fresh executor receives the MIRA work order, the fixed task prompt, and stable environment-specific instructions. The prompt identifies the available data and tools, interaction budget, editable files, submission artifact, and hidden information. The executor may write analyses and programs, run public evaluators, update its artifact, and leave evidence for later iterations. The shared outer-loop and executor interaction procedure is described in App. D; below we specify the environment-facing part of the prompt.

Each environment designates one evaluator output visible during research as its public metric; other diagnostics may be visible without determining reward. Let $q_t \in [0, 1]$ denote the normalized public quality after iteration t, with q_0 the frozen initial quality. These values define the best-so-far quality and marginal reward in Sec. 3. Each normalization is fixed before training, monotone in the designated public metric, and independent of the observed rollout cohort. After submission, a private evaluation produces a scalar private score that is never returned to the agent. It may use the same hidden target as the public metric, as in BNLearn, or private data and conditions, as in the other environments. A task is solved when its private evaluation meets the benchmark-specific success criterion.

For the three scientific environments, policy training and reported evaluation use independently initialized trajectories on the same fixed task identities and seeds. They therefore test new research trajectories and hidden within-task evaluations, not generalization to unseen task identities.

C.2 Environment interfaces

BNLearn causal discovery. We construct 100 tasks from the Alarm (32 tasks), Child (17), Insurance (24), and Water (27) networks (Scutari, 2010). Each task is a randomly permuted, anonymized 14–34-node induced subgraph. A linear-Gaussian structural equation model provides 1,800 training and 600 public-validation observations.

The agent may analyze the visible data, query validation-BIC diagnostics, and, on an n-node task, request n interventions that report estimated total effects. These diagnostics do not determine re-

ward, and graph accuracy cannot be queried during an iteration. The final artifact is an acyclic directed graph.

After each completed iteration, the harness reports directed structural Hamming distance (SHD), including missing, extra, and reversed edges. For the empty-graph SHD S_0 and submitted-graph SHD S_t,

$$q_t = \mathrm{clip}\left(\frac{S_0 - S_t}{S_0}, 0, 1\right). \tag{2}$$

The best model-authored graph is retained and restored for final scoring under the same hidden-DAG objective. Its private score is the normalized SHD gap closure above, evaluated on the submitted graph; the task is solved only by exact directed recovery.

Physics discovery. We adapt DiscoverPhysics (Wiemann et al., 2026) into 100 anonymous worlds spanning 12 mechanisms and three difficulty levels. The agent receives a mission and topology-specific laboratory interface, but not the governing law. Across 20 MIRA iterations, it may allocate 15 experiment rounds among the first 19; the last iteration is reserved for finalization. It may validate experiment plans and repeatedly fit executable candidate laws to all acquired trajectories.

The agent selects an executable law and a short explanation. For acquired-trajectory MSE m_t and position variance v_t,

$$q_t = \frac{0.95}{1 + \sqrt{m_t / v_t}}. \tag{3}$$

After the fixed horizon, the law is evaluated once on private trajectories and the explanation is graded once by a hosted GPT-5.5 judge using a hidden world-specific reference and rubric. Let n be trajectory MSE normalized by the private world's ground-truth trajectory variance, and let e be the explanation score. For an unsolved task, the private score is the product of the threshold-scaled components $0.1/\max(n, 0.1)$ and $\min(1, e/0.75)$, capped strictly below one. A task is solved, and receives a private score of one, when $n < 0.1$ and $e \geq 0.75$.

SRBench symbolic regression. We use 100 LLM-SRBench problems (Shojaee et al., 2025) from chemical reactions (30 tasks), physical oscillators (27), materials science (23), and population growth (20). Within each family, tasks span a frozen reference-expression complexity ranking. The official training data are split deterministically 80/20 into public fit and validation sets; the ID test set, optional OOD set, and reference expression remain hidden.

The agent may analyze the public data freely and evaluate expressions within a constrained SymPy-compatible grammar. It must select one final expression. The environment reports public metrics but does not restore the candidate with the highest public metric, so final selection remains with the agent.

For validation NMSE N_t, fixed initial-expression NMSE B, and float64 floor ϵ, public quality is

$$q_t = \mathrm{clip}\left(\frac{\log(B/\max\{N_t, \epsilon\})}{\log(B/\epsilon)}, 0, 1\right). \tag{4}$$

The final expression is evaluated once on sealed ID-test data, with OOD metrics reported when available. Its private score is R^2 on the sealed ID test. The task is solved when pointwise relative error is at most 0.1 everywhere (and the prediction is exact when the target is zero).

HillClimbBench-CPU architecture research. We select 50 research papers whose central architectural contribution can be implemented, trained, evaluated, and iteratively improved within a CPU-only budget, and construct one task from each paper. We run two independent trajectories per task. Each task prompt combines a technical paper summary, research brief, protected-shift description, evaluation contract, and reference implementation. The agent tests architectural hypotheses while the host fixes the data, targets, loss, optimizer, training work, evaluation seeds, and resource limits. Every candidate is trained from scratch on one CPU thread.

The 50 source papers are:

1. *Can Recurrent Neural Networks Warp Time?* (Tallec & Ollivier, 2018);

2. *An Intriguing Failing of Convolutional Neural Networks and the CoordConv Solution* (Liu et al., 2018);
3. *Deep Lattice Networks and Partial Monotonic Functions* (You et al., 2017);
4. *Deep Sets* (Zaheer et al., 2017);
5. *Learning Nonlinear Operators via DeepONet Based on the Universal Approximation Theorem of Operators* (Lu et al., 2021);
6. *Dynamical Isometry and a Mean Field Theory of RNNs: Gating Enables Signal Propagation in Recurrent Neural Networks* (Chen et al., 2018a);
7. *E(n) Equivariant Graph Neural Networks* (Satorras et al., 2021);
8. *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains* (Tancik et al., 2020);
9. *Full-Capacity Unitary Recurrent Neural Networks* (Wisdom et al., 2016);
10. *How Powerful are Graph Neural Networks?* (Xu et al., 2018);
11. *Graph Attention Networks* (Veličković et al., 2017);
12. *Grid Long Short-Term Memory* (Kalchbrenner et al., 2015);
13. *Group Equivariant Convolutional Networks* (Cohen & Welling, 2016);
14. *Learning Latent Permutations with Gumbel-Sinkhorn Networks* (Mena et al., 2018);
15. *HiPPO: Recurrent Memory with Optimal Polynomial Projections* (Gu et al., 2020);
16. *Hyena Hierarchy: Towards Larger Convolutional Language Models* (Poli et al., 2023);
17. *HyperNetworks* (Ha et al., 2016);
18. *Independently Recurrent Neural Network (IndRNN): Building A Longer and Deeper RNN* (Li et al., 2018);
19. *Interaction Networks for Learning about Objects, Relations and Physics* (Battaglia et al., 2016);
20. *The Unreasonable Effectiveness of the Forget Gate* (van der Westhuizen & Lasenby, 2018);
21. *Janossy Pooling: Learning Deep Permutation-Invariant Functions for Variable-Size Inputs* (Murphy et al., 2018);
22. *KAN: Kolmogorov-Arnold Networks* (Liu et al., 2024);
23. *Legendre Memory Units: Continuous-Time Representation in Recurrent Neural Networks* (Voelker et al., 2019);
24. *Resurrecting Recurrent Neural Networks for Long Sequences* (Orvieto et al., 2023);
25. *Long Short-Term Memory* (Hochreiter & Schmidhuber, 1997);
26. *Mamba: Linear-Time Sequence Modeling with Selective State Spaces* (Gu & Dao, 2023);
27. *Memory Networks* (Weston et al., 2014);
28. *MinimalRNN: Toward More Interpretable and Trainable Recurrent Neural Networks* (Chen, 2017);
29. *Hopfield Networks is All You Need* (Ramsauer et al., 2020);
30. *Neural Arithmetic Units* (Madsen & Johansen, 2020);
31. *Neural Ordinary Differential Equations* (Chen et al., 2018b);
32. *Neural Processes* (Garnelo et al., 2018);
33. *Neural Turing Machines* (Graves et al., 2014);
34. *OptNet: Differentiable Optimization as a Layer in Neural Networks* (Amos & Kolter, 2017);
35. *Parallelizing Legendre Memory Unit Training* (Chilkuri & Eliasmith, 2021);
36. *PDE-Net: Learning PDEs from Data* (Long et al., 2017);
37. *Perceiver IO: A General Architecture for Structured Inputs & Outputs* (Jaegle et al., 2021);

38. *Rethinking Attention with Performers* (Choromanski et al., 2020);
39. *Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences* (Neil et al., 2016);
40. *Pointer Networks* (Vinyals et al., 2015);
41. *Recurrent Highway Networks* (Zilly et al., 2016);
42. *Efficiently Modeling Long Sequences with Structured State Spaces* (Gu et al., 2021);
43. *Orthogonal Recurrent Neural Networks with Scaled Cayley Transform* (Helfrich et al., 2017);
44. *Implicit Neural Representations with Periodic Activation Functions* (Sitzmann et al., 2020);
45. *Object-Centric Learning with Slot Attention* (Locatello et al., 2020);
46. *Spatial Transformer Networks* (Jaderberg et al., 2015);
47. *TabNet: Attentive Interpretable Tabular Learning* (Arik & Pfister, 2019);
48. *Less is More: Recursive Reasoning with Tiny Networks* (Jolicoeur-Martineau, 2025);
49. *Universal Transformers* (Dehghani et al., 2018); and
50. *Zoology: Measuring and Improving Recall in Efficient Language Models* (Arora et al., 2023).

The public evaluator measures both in-distribution performance and a task-specific structural extrapolation split. Each task fixes in advance a scoring function that combines these measurements into a scalar public metric $u_t \in [0, 1]$, with one denoting the task's declared ideal. Only valid candidates whose public metric exceeds that of the retained incumbent replace it. For the frozen reference metric u_0,

$$q_t = \mathrm{clip}\left(\frac{u_t - u_0}{1 - u_0}, 0, 1\right). \tag{5}$$

After selection, the retained architecture and task reference are retrained on the same five private seeds under harder structural conditions. The task's private evaluator maps each run to a score in $[0, 1]$, and the submission's private score is the mean across its five runs. A submission meets the benchmark's success criterion when the lower endpoint of the two-sided 95% Student-t confidence interval over the five paired candidate-minus-reference scores is positive. In the policy comparison, a win is a paired evaluation in which the trained-policy submission has a higher private score than the matched untrained-MIRA submission.

C.3 Evaluation

We report the public best-so-far quality $\bar{q}_t$ and the private score separately. Comparisons use matched task instances and interaction budgets; confidence intervals use a task-clustered bootstrap.

D Additional MIRA Details

This appendix describes how MIRA constructs each decision context, executes the resulting work order, and delegates evaluation and submission to the task environment.

D.1 Context construction

The outer-loop context is rebuilt at every iteration. Its fixed portion contains the task statement, a compact workspace summary, the latest public evaluation, a structured summary of the immediately preceding work episode and a pointer to its handover when available, and the remaining budget. The full episode record and handover text are not copied automatically. Instead, the policy has read-only access to repository files, diffs, commit history, and prior work records, from which it curates a bounded set of decision-relevant evidence. The fixed portion and this selected evidence form the new decision context.

The policy is instructed to select a bounded objective or coherent experiment set while considering its information value, cost, remaining time, and risk to the current solution. Its work order states the

objective, relevant evidence, resource guidance, acceptance criteria, and stopping condition. The same structure is used for theorem proving, with experiments replaced by proof construction and verification.

D.2 Decision and execution

A fresh Codex inner loop receives the work order, its selected repository references, and the stable environment instructions. It may inspect and modify the workspace, run tests and authorized tools, and evaluate candidates. It validates substantive work, commits useful artifacts and evidence, and leaves a concise handover. Recorded failures and contradictory results remain available to later decisions through the repository.

The repository—including the current candidate, experiments, results, analyses, and handovers—persists across iterations; the model contexts do not. Any public evaluation produced after execution becomes evidence for the next decision. When an environment maintains an incumbent, it protects the strongest validated artifact against later regressions.

D.3 Environment-controlled evaluation and submission

The environment adapter supplies the task-specific prompt, tools, public feedback, reward mapping, submission artifact, and terminal evaluator. It also determines who authorizes submission: Codex may submit under inner-loop control, the policy chooses between further investigation and submission under outer-loop control, and the environment evaluates only after research ends under external control.

A run ends when a model-visible success condition is met, submission is authorized, or the iteration or wall-clock budget is exhausted. Private evaluation remains outside both model contexts and is applied only to the selected repository artifact. Environments that retain an incumbent may restore it before final evaluation.

E Semantic Exploration Graphs and Strategic Inertia

We construct the exploration graphs after each run has terminated. The graphs are used only for analysis and are never shown to the evaluated system.

Comparable evidence projection. We project every run into the same chronological evidence format: the common task statement, text authored by the evaluated models, and the submitted proof. For MIRA, model-authored text includes the outer agent's work orders and the inner agents' reports; for Codex, it includes assistant messages. We exclude commands, tool outputs, diffs, commits, repository metadata, handovers, submission reviews, and external evaluations from every condition. The same projection is applied at each forced-continuation checkpoint, and the same downstream extraction procedure is applied to every inference system.

Semantic graph extraction. A GPT-5.5 pass at temperature zero converts the projected trace into a graph. A node represents a semantically meaningful proof or research step, such as an approach, claim, lemma, counterexample, experiment, candidate, result, or submission. Several raw turns may therefore map to one node, while one long response may contain several nodes. Nodes record their status, such as supported, proved, refuted, stalled, abandoned, or submitted. Typed edges record evidence-supported relationships including refinement, dependence, support, refutation, pivoting, supersession, and production of a result. Chronology alone does not create an edge.

The extractor receives the trajectory content but not the method name, model name, grader feedback, or final grader score. Every node and edge must cite a source segment in the projected evidence; outputs without valid provenance are rejected. The extracted nodes are ordered by first appearance in the trace. For visualization, the submission ancestry is placed on a central lane and other connected components are assigned separate lanes. Shortened labels and display coordinates affect only presentation, not the analysis.

Research-direction annotation. A separate GPT-5.5 pass at temperature zero groups the fixed semantic nodes into research directions. A direction is a high-level hypothesis, proof plan, construction, mechanism, or algorithmic approach. Lemmas, tests, repairs, and refinements remain in the same direction when they serve the same high-level plan; a materially different plan receives a new identifier, and a later return to a previous plan reuses its identifier. This pass may correct the order of fixed nodes when their provenance establishes a different first-introduction order, but it may not add, remove, split, or merge nodes. It observes node content, provenance episode identifiers, and graph relations, but not the method or model name, final grader score, grader feedback, node status, terminal submission, graph title, or graph summary.

Strategic inertia. Two consecutive non-submission semantic steps form a continuation when they belong to the same annotated research direction and a pivot otherwise. For graph $\mathcal{G}$, we define $I(\mathcal{G}) = N_{\text{continue}}/(N_{\text{continue}} + N_{\text{pivot}})$. The root and terminal submission are excluded. Graphs with fewer than two non-submission nodes have no transitions and receive no inertia value. The statistic is descriptive: high inertia can sustain a promising direction or prolong a local minimum, while low inertia can expose alternatives or reflect premature switching.

For the natural GPT-5.5 comparison, the analysis contains 30 problems, three trials, and three inference systems. We compute inertia for each run, average trials within each problem, and obtain 95% confidence intervals by resampling problems with 10,000 bootstrap draws. Pairwise contrasts match runs by problem and trial, average the paired differences within each problem, and again bootstrap problems. Natural and forced-continuation runs are analyzed separately.

The four visualizations below are the problems on which a gap remains after output-token-matched continuation. They show the first natural-stopping trial and use the same extraction and layout procedure for all three inference systems.

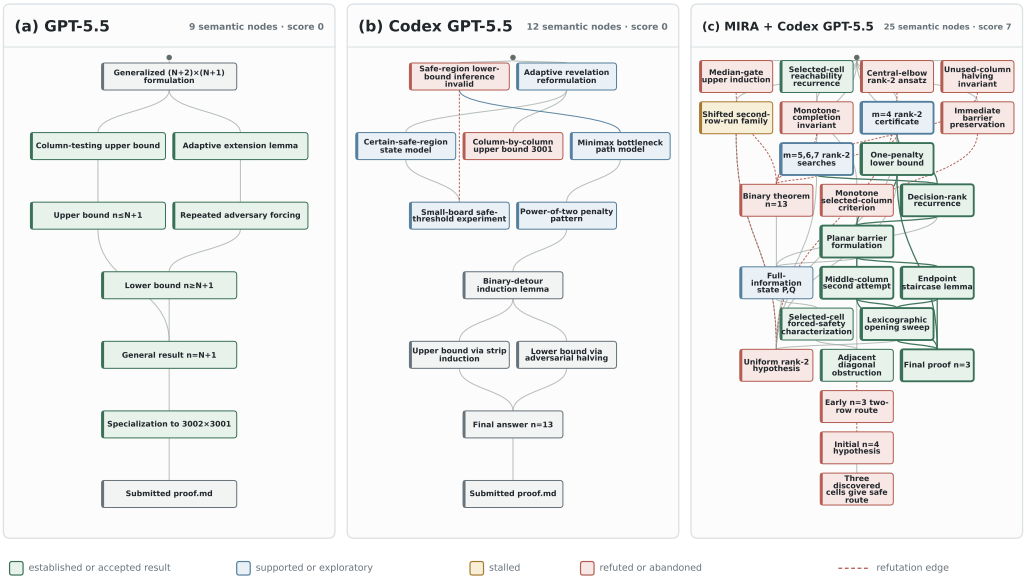


Figure 8: Semantic exploration graphs on PB-Advanced-023. Panels show (a) GPT-5.5, (b) Codex GPT-5.5, and (c) MIRA with Codex GPT-5.5. MIRA explores multiple paths and ultimately recovers the correct proof. Dashed red edges denote refutations.

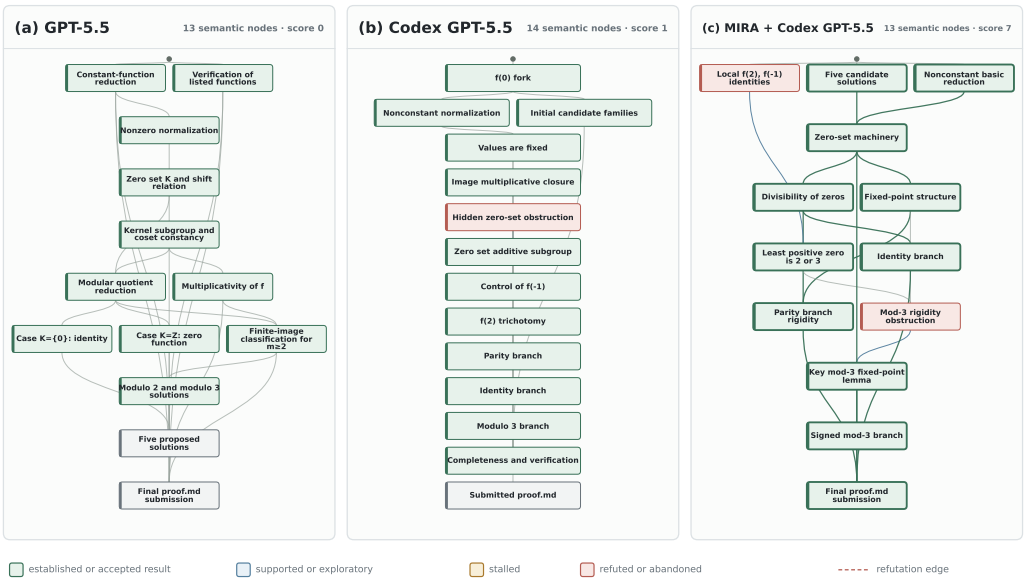


Figure 9: **MIRA develops and combines multiple proof branches into a complete classification on PB-Advanced-006.** GPT-5.5 and persistent Codex identify candidate functions but leave central steps unsupported or incomplete, whereas MIRA produces a complete 7/7 proof. Systems and notation follow Figure 8.

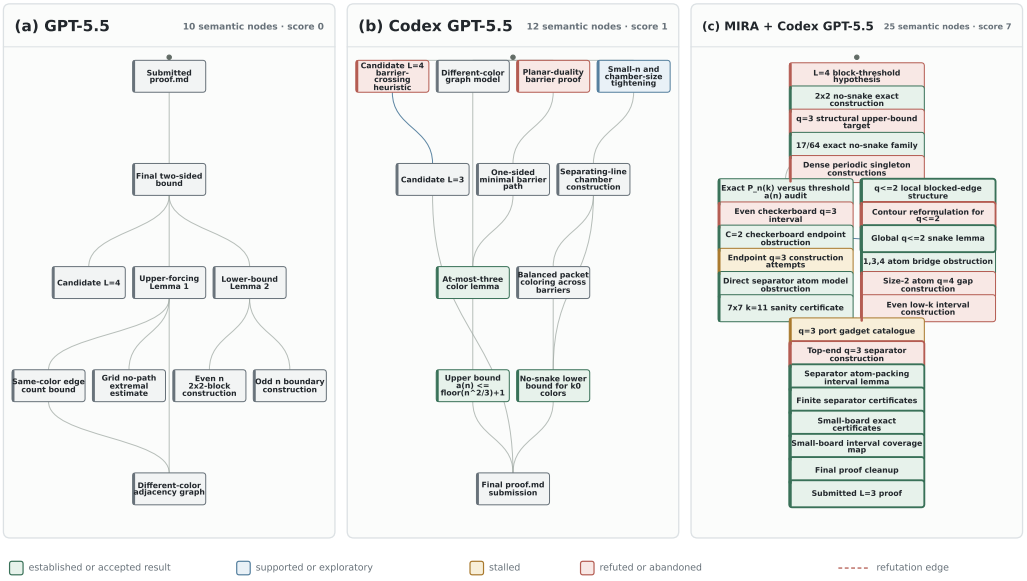


Figure 10: **MIRA redirects from a refuted $L = 4$ approach and completes an $L = 3$ proof on PB-Advanced-018.** GPT-5.5 remains within the unsuccessful $L = 4$ direction, while persistent Codex pivots to $L = 3$ but leaves the proof incomplete. MIRA develops the redirected approach into a complete 7/7 proof. Systems and notation follow Figure 8.

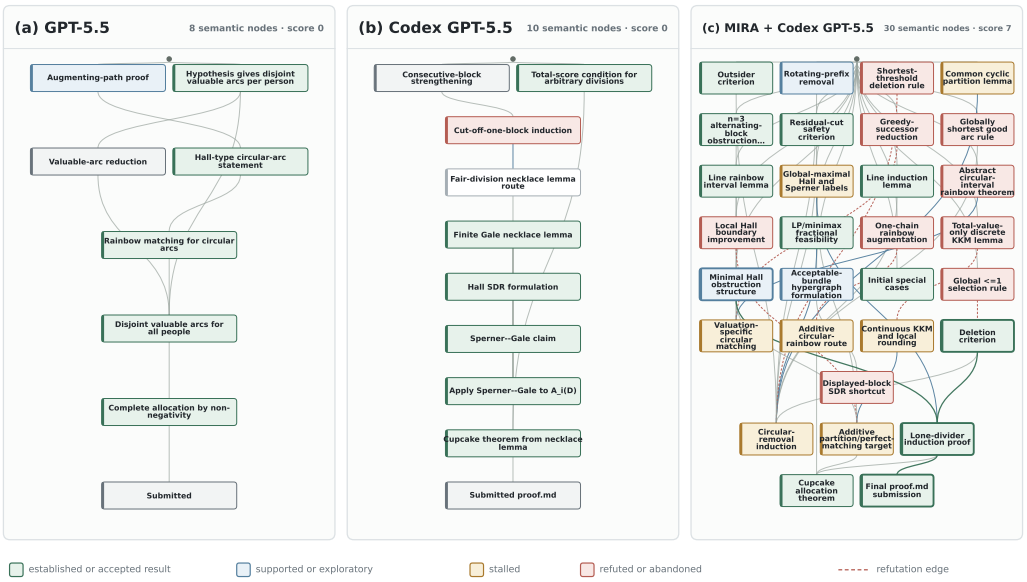


Figure 11: **MIRA explores competing combinatorial directions and synthesizes a correct proof on PB-Advanced-030.** GPT-5.5 and persistent Codex terminate in unsuccessful proof directions, whereas MIRA explores allocation, deletion, and induction arguments before reaching a complete 7/7 proof. Systems and notation follow Figure 8.

F RMT Architecture Identified by MIRA

Matched search protocol. We run one 24-hour GPT-5.5 campaign for each orchestration condition. All three campaigns use the same task reference architecture, public evaluator, compute budget, and parameter limit. Public evaluations guide research, and the final submitted architecture is scored separately by the private evaluator.

Selected architecture. An ordinary RMT stores each token's residual state as a matrix $X_\ell \in \mathbb{R}^{D_k \times D_v}$. Attention and MLP branches read this state through a lower-rank interface, produce a matrix update $\Delta X_\ell^{(b)}$, and add it uniformly as $X_{\ell,\text{out}}^{(b)} = X_{\ell,\text{in}}^{(b)} + \Delta X_\ell^{(b)}$ for $b \in \{\text{attn}, \text{mlp}\}$. MIRA's final model retains this matrix state and low-rank read/write interface, uses a GEGLU feed-forward block, and introduces a separate learned gate for each branch and layer: $X_{\ell,\text{out}}^{(b)} = X_{\ell,\text{in}}^{(b)} + G_\ell^{(b)} \odot \Delta X_\ell^{(b)}$, where $G_\ell^{(b)} \in \mathbb{R}^{D_k \times D_v}$. Every coefficient is initialized to one, so the new update is exactly the ordinary additive update at initialization. The two gates add only 0.056% to the model's parameter count. Unlike a generic GEGLU block, this mechanism is specific to the matrix residual stream: it learns where each branch can amplify or suppress a proposed matrix-state write.

MIRA first tested the gate at small scale, then reran a geometry-, seed-, optimizer-, and token-matched comparison at the final scale. Relative to the ungated GEGLU control, the gated model improved last-token cross-entropy by 0.009 at small scale and by 0.021 at the final scale. Both comparisons use one seed, and the ungated final-geometry control was not submitted to the private evaluator. These ablations support the gate as a promising mechanism, but they do not isolate it as the cause of the full private-loss improvement in Figure 2(b).

The learned gates did not remain near their identity initialization. Attention gates generally amplify writes, whereas MLP gates generally suppress them, and both vary with depth. Additive row-plus-column effects explain roughly three quarters of the variation in the depth-averaged attention map and nearly nine tenths in the MLP map, with stronger variation along the residual-key axis. This supports a structured, branch-, depth-, and address-dependent *write-allocation* interpretation. It does not establish dynamic routing: the gates are fixed across tokens and examples, and their low

exact-map correlation across layers does not support a clean permanent partition into "durable" and "temporary" cells.

G Loop Transformer Search and Selected Designs

Matched search protocol. We compare one 48-hour MIRA campaign with one persistent Codex campaign using GPT-6 Astra. Both campaigns use the same model-scale and training-token rungs. At each rung, candidates and the task reference architecture are evaluated on the same 512-context public-proxy set with eight recurrent iterations. Intermediate search decisions use these results; after submission, the final incumbent from each campaign and the task reference are evaluated on a separate 4,873-context private set. The public and private results are reported separately in Figure 4.

MIRA search trajectory. MIRA investigates explicit fast and slow states, iteration-history mechanisms, and contrastive attention operators. It uses matched controls to test whether apparent improvements come from the proposed architectural change rather than added capacity, and larger validation sets to reject gains that do not persist. The selected design combines signed attention with direct retention of the previous recurrent state. If the two branches produce attention reads A_1 and A_2 over shared values, the attention module returns $A_1 + (A_1 - A_2)$. Each branch separately uses ordinary softmax attention; "signed" refers to the effective positive and negative coefficients created by subtracting the two reads. The second branch copies the first branch's query and key parameters at initialization, making the additional contrastive term exactly zero before training. It can then diverge to identify content that should be cancelled rather than reinforced on subsequent passes through the shared block. After the block proposes its next state $F(h_t)$, a fixed residual mixture preserves a direct path from the prior state, $h_{t+1} = \rho h_t + (1 - \rho)F(h_t)$, with $\rho = 0.25$. Signed attention and state retention therefore play complementary roles: one lets the recurrent computation revise or suppress information, while the other prevents repeated updates from erasing useful state. This adapts the contrastive mechanism of Differential Attention (Ye et al., 2025) to the distinct setting of a weight-tied recurrent model.

Comparison. Persistent Codex selects a tangent-state update with local/global attention routing and learned write gates on both the attention and MLP branches, a complementary strategy that regulates how aggressively each recurrent pass changes the state. MIRA's incumbent improves public-proxy loss at every completed scale. Its final public proxy is 3.0331, compared with 3.2419 for Codex and 3.2940 for the task reference. On the separate private evaluation, MIRA scores 3.0464, compared with 3.2275 for Codex and 3.2653 for the task reference.

H Critic Details

Targets and value contracts. All critic variants use the stitched Monte Carlo target G_t. Under public marginal supervision, G_t is the future increase in best-so-far public quality, where q_t is defined for each environment in App. C. The corresponding public qualities are directed-SHD gap closure for BNLearn, acquired-trajectory normalized fit for Physics Discovery, validation log-NMSE gap closure for SRBench, and initial-to-ideal public-score gap closure for HillClimbBench-CPU.

Under private-binary supervision, the label $b_t \in \{0, 1\}$ records whether the final submission meets the environment's private success criterion. This means exact directed recovery for BNLearn; the joint hidden-trajectory and explanation criterion for Physics Discovery; the ID-test pointwise-accuracy criterion for SRBench; and a positive lower endpoint of the paired 95% confidence interval for improvement over the reference architecture for HillClimbBench-CPU. These are the same success criteria specified in App. C; the policy-comparison win rate is not a critic target.

Critic parameterizations and objectives. For public marginal targets, a scalar critic minimizes

$$\mathcal{L}_{\text{scalar}}(\phi) = \frac{1}{M} \sum_{m=1}^{M} \left(V_\phi(o_m) - G_m \right)^2 . \tag{6}$$

For the categorical critics, let $z_1, \ldots, z_K$ be the ordered support. If $z_j < G_t < z_{j+1}$, we project G_t onto its neighboring bins as

$$y_j(G_t) = \frac{z_{j+1} - G_t}{z_{j+1} - z_j}, \qquad y_{j+1}(G_t) = \frac{G_t - z_j}{z_{j+1} - z_j}, \tag{7}$$

with all other entries zero; if $G_t = z_j$, then $y_j(G_t) = 1$. A categorical head produces $q_\phi(z_i \mid o) = \mathrm{softmax}(h_\phi(o))_i$, while the generative critic uses the restricted next-token distribution $p_\phi(z_i \mid o)$ from Sec. 3.1. They minimize

$$\mathcal{L}_{\mathrm{cat}}(\phi) = \frac{1}{M} \sum_{m=1}^{M} D_{\mathrm{KL}}(y(G_m) \,\|\, p_\phi(\cdot \mid o_m)) \,, \tag{8}$$

replacing p_ϕ with q_ϕ for the categorical head.

Under private-binary supervision, the scalar critic outputs a single logit $\ell_\phi(o)$ and represents value as $V_\phi(o) = \sigma(\ell_\phi(o))$. It is trained with binary cross-entropy with logits,

$$\mathcal{L}_{\mathrm{bin}}(\phi) = -\frac{1}{M} \sum_{m=1}^{M} \left[b_m \log \sigma(\ell_\phi(o_m)) + (1 - b_m) \log(1 - \sigma(\ell_\phi(o_m))) \right] . \tag{9}$$

The private-binary generative critic instead restricts the next-token distribution to the support $\{0, 1\}$ and uses the corresponding one-hot target. Token-level variants repeat the decision return at each causal token prefix within the action; step-level variants predict once from the curated observation.

Target and prediction validation. Public marginal returns are validated and clipped to $G_t \in [0, 1 - \bar{q}_{t-1}]$. Before bootstrapping, temporal-difference targets, or advantage estimation, public marginal predictions must be finite and are clipped to the same interval. For a generative critic, $V_\phi(o_t) = \sum_i p_\phi(z_i \mid o_t) z_i$. Private-binary targets lie in $\{0, 1\}$, and both the sigmoid head value and the expected value of the two-point generative distribution lie in $[0, 1]$.

Pretraining and target-environment adaptation. We pretrain Qwen-27B for one epoch with batch size 8, microbatch size 1, maximum sequence length 262,144, and seed 0. The optimizer is AdamW with learning rate 10^{-5}, weight decay 0.01, gradient clipping at 1.0, 5% linear warmup, and cosine decay to 10% of the peak learning rate. Training uses bfloat16 and rank-32 LoRA adapters (scale 64, dropout 0) on four H200s. For target-environment adaptation, we evaluate support sizes $\{0, 1, 5, 10, 25, 50\}$ over five paired support/query splits with 50 query episodes per split. Adaptation makes one episode-ordered pass using AdamW at learning rate 10^{-5} and LoRA dropout 0.05 on one H200.

For each target environment and paired support/query split, we first average the single-source pretraining condition across source environments. We then report averages over target environments and splits with paired-seed bootstrap 95% confidence intervals.

Generative critic prompts. The generative critic receives the same curated decision-scoped conversation as the actor, followed by the value query in Figure 12. During terminal prediction, the query marks the normal research budget as exhausted.

```
Value contract: <ENVIRONMENT-SPECIFIC VALUE-CONTRACT NAME>
<ENVIRONMENT-SPECIFIC VALUE-CONTRACT DESCRIPTION>

Observable value state:
Current best normalized quality before this decision: <CURRENT BEST NORMALIZED QUALITY>
Total decision horizon: <TOTAL DECISION HORIZON>
Decisions remaining including the current decision: <DECISIONS REMAINING>
<OPTIONAL TERMINAL-PHASE STATUS>

You are estimating the value of the research state in the conversation above.

Predict the future increase in the final best solution quality, using the normalized 0-to-1 quality scale.

0.0 means no further improvement over the best solution already found.
Higher values mean larger expected improvement in final best solution quality.

Use the raw metrics, errors, tests, benchmark scores, and experiment results as evidence, but do not answer in raw environment units.

On this normalized 0-to-1 scale, how much additional improvement over the current best solution do you expect from here?
A. 0.0
B. 0.1
C. 0.2
D. 0.3
E. 0.4
F. 0.5
G. 0.6
H. 0.7
I. 0.8
J. 0.9
K. 1.0

Answer with only the single letter for your estimate.
```

Figure 12: **Canonical prompt for eliciting public-marginal generative value forecasts.** Environment-specific fields specify the value contract, current normalized quality, and remaining budget. Although the model emits a single letter, its next-token probabilities over all labels define the critic's distribution over remaining return.

For private-binary supervision, the environment-specific contract states the private success event above, and the public-marginal question and eleven-point support are replaced by:

```
Predict the binary terminal outcome of the complete research episode.

0.0 means the environment's authoritative terminal success condition will not be met.
1.0 means the environment's authoritative terminal success condition will be met.

Use the raw metrics, errors, tests, benchmark scores, experiment results,
current progress, and remaining budget as evidence.

Will this research episode ultimately succeed?
A. 0.0
B. 1.0

Answer with only the single letter for your estimate.
```

I Implementation Details

All policy-training runs use Qwen-27B with rank-32 LoRA adapters (scale 64, dropout 0). Separate actors target the standard attention and MLP projections, MoE experts, and latent fully connected projections. MIRA-AC targets the standard attention and MLP projections together with the model's recurrent input and output projections. Separate-critic variants use distinct policy and critic adapters. In MIRA-AC, one shared adapter serves both roles, with separate optimizer states for actor and critic updates; online critic updates train only its MLP LoRA parameters. Each critic is initialized from the corresponding critic-parameterization- and target-environment-specific pretrained checkpoint. Separate actors start from the untrained MIRA policy. For each single-environment experiment, MIRA-AC initializes its shared adapter from a generative critic pretrained on trajectories from that environment under the matching supervision target. The cross-environment policy instead starts from a public-marginal generative critic pretrained jointly across all four environments, then receives online actor–critic updates on SRBench, Physics Discovery, and HillClimbBench-CPU.

We train one policy per configuration. Each run makes one pass over a fixed ordered set of 100 episodes with task-order seed 42. The scientific environments use 100 tasks with one rollout each; HillClimbBench-CPU uses 50 tasks with two rollouts each. Episodes permit at most 20 research decisions. The maximum sequence length is 262,144 tokens, and meta-reasoning generations are capped at 8,192 tokens with temperature 1.0, top-p 0.95, and top-k 20. Policy generation enables the model's thinking mode; critic prediction disables it.

The actor uses FP32 AdamW with learning rate 10^{-5}, $\beta = (0.9, 0.999)$, weight decay 0.01, gradient norm clipping at 1.0, and a constant schedule. Online critic updates use a bfloat16 backbone, with logits and targets cast to FP32 for the loss. The critic uses AdamW with learning rate 5×10^{-6}, weight decay 0.01, gradient norm clipping at 1.0, microbatch size 1 across four GPUs, and one optimizer step after a complete without-replacement pass over each sealed batch. The public generative critic uses support $\{0.0, 0.1, \ldots, 1.0\}$; the private-binary critic predicts the binary private outcome on support $\{0, 1\}$. We use $\gamma = 1$ and exact Monte Carlo critic targets. Actor advantages use episode-length-adaptive GAE with $\lambda_T = 1 - 1/(\alpha T)$ and $\alpha = 20$ for an episode of T decisions.

Let $a_{t,j}$ be token j of decision a_t, L_t its number of sampled tokens, and θ_{b_t} the actor version that generated it. Define

$$\rho_{t,j} = \frac{\pi_\theta(a_{t,j} \mid o_t, a_{t,<j})}{\pi_{\theta_{b_t}}(a_{t,j} \mid o_t, a_{t,<j})}, \qquad m_{t,j} = \mathbf{1}[1 - \epsilon_{\text{low}} < \rho_{t,j} < 1 + \epsilon_{\text{high}}]. \tag{10}$$

For a batch of N decisions, the implemented decision-normalized actor objective is

$$\mathcal{L}_{\text{actor}}(\theta) = \frac{1}{N} \sum_{t=1}^{N} \frac{1}{L_t} \sum_{j=1}^{L_t} \left[-m_{t,j} \tau_A \hat{A}_t \rho_{t,j} + \tau_{\text{KL}} (\log \rho_{t,j})^2 \right]. \tag{11}$$

We set $\epsilon_{\text{low}} = 0.3$, $\epsilon_{\text{high}} = 5.0$, $\tau_A = 1$, and $\tau_{\text{KL}} = 10^{-3}$. Tokens outside the ratio interval are excluded from the policy-gradient term but remain in the squared-log-ratio penalty. The trainable span contains only the sampled work-order or stop response, including its end-of-turn token when present; prompts, curated observations, tool calls, and executor outputs are masked.

The buffered schedule launches rollouts in groups of 16 and maintains at most 32 active episodes. Every completed episode enters both buffers. The actor buffer triggers when completed episodes jointly reach four episodes, 64 decisions, and 262,144 actor tokens; after 15 minutes, it may trigger once the episode and decision minima are met. The critic buffer independently triggers at two episodes, 32 decisions, and 131,072 prompt tokens, with a 7.5-minute fallback after its episode and decision minima are met. At the end of training, any remaining completed episodes form final underfilled batches. In MIRA-AC, role-specific losses update the shared adapter sequentially; if both buffers are ready, the critic update precedes the actor update, and recorded values remain detached during actor training. Either loss publishes the next shared model version, which live episodes adopt only at the next meta-reasoning boundary.

Critic pretraining uses four H200 GPUs. Actor–critic training uses two nodes with eight H200 GPUs each.

J Trajectory Stitching Across Artificial Boundaries

MIRA reconstructs a fresh curated context at every MIRA iteration. Context-window limits, compaction, sequence packing, and learner batching may further split a single logical episode into multiple samples. None of these boundaries is an environment termination. Treating them as terminal either removes future TD residuals from earlier advantages or places later rewards at the wrong temporal distance.

Let a logical episode $\tau = (o_1, a_1, r_1, \ldots, o_T, a_T, r_T, o_{T+1})$ be partitioned into stored samples $C_1, \ldots, C_K$. The partition does not alter the termination indicator: $d_t = 1$ only when the environment truly terminates. Each stored transition contains o_t, a_t, r_t, o_{t+1}, actor behavior log probabilities and version θ_{b_t}, and the critic version c_t assigned to the transition. That critic records both the current baseline $v_t = V_{\phi_{c_t}}(o_t)$ and its successor bootstrap $\tilde{v}_{t+1}^{(c_t)} = V_{\phi_{c_t}}(o_{t+1})$.

An asynchronous episode may adopt a different critic before decision $t + 1$. At a nonterminal boundary where this occurs, let

$$B_t = v_{t+1} - \tilde{v}_{t+1}^{(c_t)}, \qquad v_{t+1} = V_{\phi_{c_{t+1}}}(o_{t+1}), \tag{12}$$

and set $B_t = 0$ otherwise. Terminal successor values and bridges are zero. The implemented residual and backward recurrence are

$$\delta_t = r_t + \gamma(1 - d_t)\tilde{v}_{t+1}^{(c_t)} - v_t, \qquad \hat{A}_t = \delta_t + \gamma\lambda_T(1 - d_t)(\hat{A}_{t+1} + B_t). \tag{13}$$

The bridge aligns the outgoing critic's successor baseline with the baseline recorded by the critic version assigned to the next decision.

After this episode-level pass, $\hat{A}_t$ is scattered back to its original sample and token span for actor training. The critic target is the Monte Carlo return G_t, computed separately over the same reconstructed episode. The recorded value forecasts are treated as constants during the actor update, so the policy loss does not differentiate through the shared critic baseline.

Partition invariance. For any partition of a fixed logical episode into stored samples, stitching produces the same actor advantages and Monte Carlo critic targets as processing the unpartitioned episode, provided the logged transitions, critic versions, and successor bootstraps are preserved. Stitching restores the same ordered rewards, terminal indicators, baselines, and cross-version bridges. A backward induction on the recurrence therefore yields the same $\hat{A}_t$ at every decision, while every reward remains in the same Monte Carlo return.

In our implementation, $\gamma = 1$ and the critic is trained on the exact Monte Carlo return $G_t = \sum_{u=t}^{T} r_u$. Actor advantages use episode-length-adaptive $\lambda_T = 1 - 1/(\alpha T)$, giving $\hat{A}_t = \delta_t + \lambda_T(1-d_t)(\hat{A}_{t+1} + B_t)$. The full Monte Carlo return receives weight $\lambda_T^{T-1} \geq 1 - 1/\alpha$ at the earliest decision and greater weight at later decisions. The bridges correct baseline-coordinate changes but do not alter the Monte Carlo critic target. Stitching remains necessary: G_t must include rewards from all later MIRA iterations, even when those iterations occupy different contexts, samples, or learner batches.

The governing invariant is that only true environment termination may stop value bootstrapping; infrastructure boundaries never do.

K Additional MIRA-AC Results

L Behavioral Analysis of MIRA-AC

We compare the cross-environment MIRA-AC policy with untrained MIRA on 400 matched evaluation pairs (800 trajectories): 100 pairs each from symbolic regression, physics discovery, HillClimbBench-CPU, and BNLearn. For each pair, a judge first compared how the two agents conducted research without seeing their policy identities or protected evaluation outcomes. We froze this assessment before a second pass showed the judge each trajectory's protected outcome and asked which outer policy had contributed more to the final result. Policy identity remained hidden in both passes. This ordering lets the second pass interpret exploratory choices in light of whether they paid off without allowing the outcome to rewrite the initial description of the research process.

Figure 14 pools all four environments. Of the 334 pairs for which the judge made a decisive attribution, it favored MIRA-AC in 203 (60.8%) and untrained MIRA in 131; the remaining 66 pairs were tied or unclear. Relative to untrained MIRA, MIRA-AC more often took a defensible risk that improved the protected result (52.5% vs. 34.8%), reconsidered the underlying approach (46.3% vs. 37.0%), and ran tests that distinguished between competing ideas (28.0% vs. 16.8%). It also more often checked whether an improvement held up (7.0% vs. 2.5%) and stopped when further work appeared unlikely to help (13.3% vs. 7.3%), while less often spending its remaining budget on minor variations of the same unproductive idea (18.8% vs. 33.5%).

These are judge-coded associations and attributions, not interventionally identified causal effects of any single behavior. They therefore support the claim that training changes the observed research process, but do not by themselves establish that independently inducing one listed behavior would improve performance.

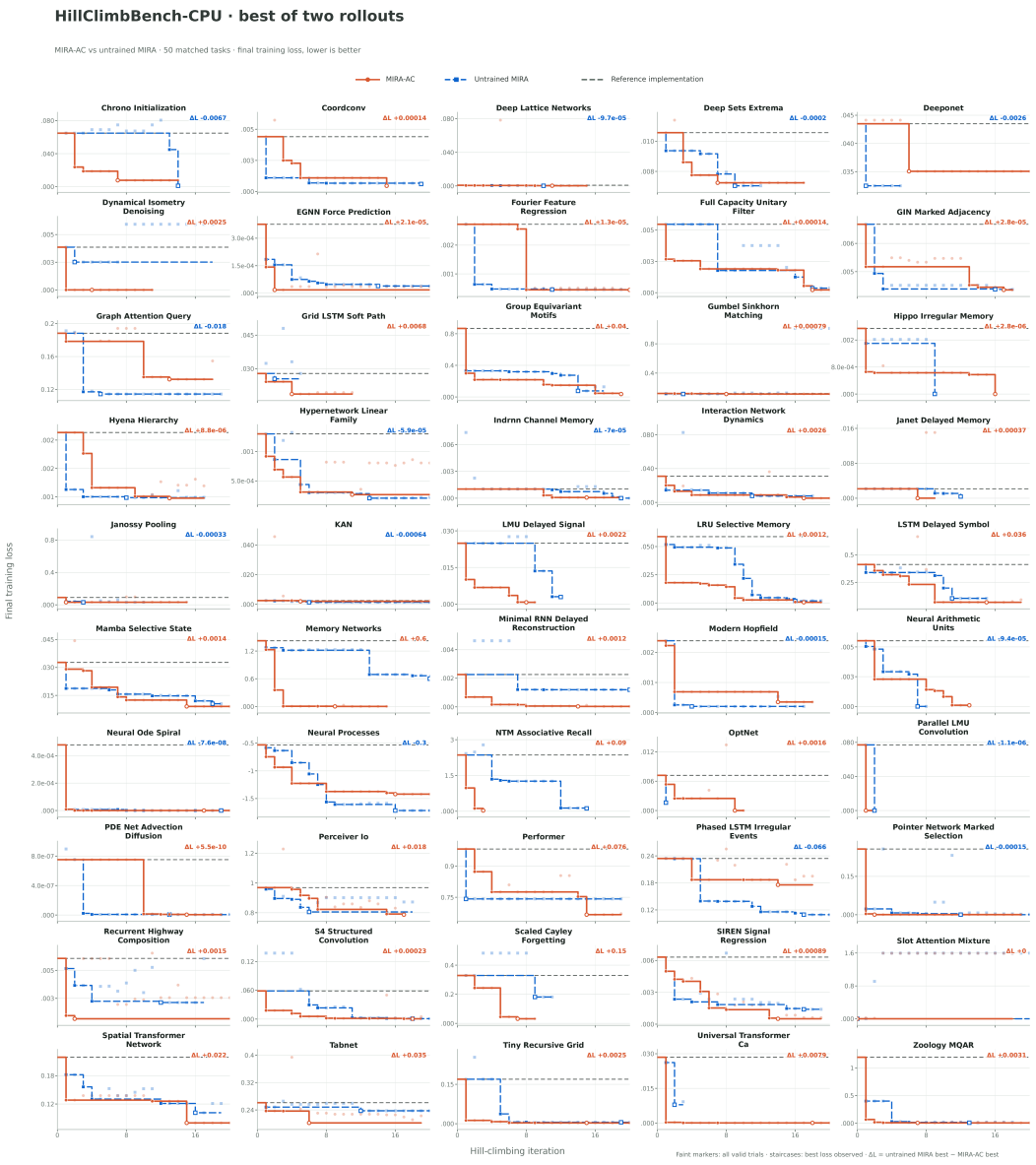
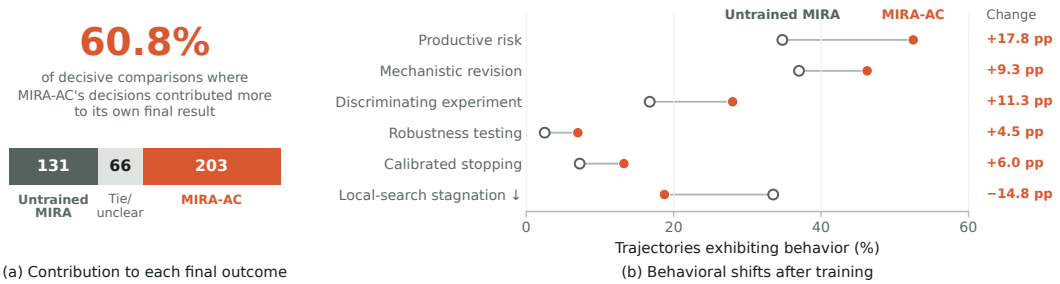


Figure 13: **MIRA-AC achieves a lower best-observed loss than untrained MIRA on 33 of 50 HillClimbBench-CPU task families.** Each panel compares the trajectory with the lower best-observed raw loss from two matched rollouts per policy. Faint points show successful evaluations, staircases track the best loss observed so far, and the horizontal dashed line marks the reference implementation.



(a) Contribution to each final outcome

(b) Behavioral shifts after training

Figure 14: **MIRA-AC changes how the agent conducts research.** Results pool 400 matched comparisons across four environments. **(a)** Which outer policy the outcome-aware judge credited with contributing more to the protected result. **(b)** The fraction of trajectories exhibiting each research behavior; arrows mark the direction considered preferable.