Dynamical Synthesis

Learning through Interaction

Brian Ashiundu*

Research Fellow, Wolfram Institute

Head of Research, Dilate Technologies

brianashiundu000@gmail.com bmboya@wolframinstitute.org

11 September 2026

Abstract

How can a computational system learn from continuing interaction, retain useful capabilities and revise its behaviour when new evidence challenges earlier conclusions? We introduce Dynamical Synthesis, a design framework for describing how internal computation and external input contribute to changes in a system and what it learns. External input includes observations from environments, people and other systems. We describe a computational model that separates changes during interaction from training changes to model parameters, and specifies how proposed updates should be checked. We test selected parts through controlled studies and examine records from a language-model implementation. In a controlled mathematics comparison, additional training produced 43 accepted answers out of 64 questions using new numerical values in problem types represented in training; the base model without additional training and a control trained on shuffled question-answer pairs each produced none. Acceptance required exact mathematical checks and a separate AI review. Follow-up studies found no advantage over the shuffled control on harder mathematics, and further response training produced more losses than gains in answer performance. These findings show improvement in a specific training comparison, with limits on applying learning to new problems and preserving earlier abilities. Whether the complete framework supports lasting learning beyond its individual parts remains to be tested.

1 Introduction

An artificial system that continues learning raises a design question: how can interaction with its environment produce useful, lasting changes while keeping the system open to evidence that challenges what it has learned? The environment may supply measurements, results of actions, human instruction or messages from other systems. A learner may improve on a task while losing earlier abilities or repeatedly confirming an error through its own feedback. We therefore need to examine both the information supplied through interaction and the changes it produces.

Human oversight is one application of this broader question. We use *alignment* to describe the problem of relating system behaviour and goals to legitimate human intentions and interests. Several people may be affected, and their interests may differ. Following one participant's next instruction is therefore an incomplete test of alignment. One approach, called cooperative inverse reinforcement learning (CIRL), models a shared human-agent goal that the agent initially does not know fully. The model states its assumptions about what the agent can learn from the human. (Hadfield-Menell et al., 2016).

Loopseed is a research programme investigating whether computational systems can learn through continuing interaction, retain useful changes and remain responsive to correction. We call its organising idea **Dynamical Synthesis**:

$$I = W(I) + \text{You}. \tag{1}$$

*Research conducted under Dilate Technologies.

Here I means system activity and W its internal processing. **You means input received from outside a defined system boundary**. It includes people, other agents and physical or simulated environments; it need not use language or come from a source with intentions. The boundary states what counts as part of the system: a model, an agent with memory and tools, or a network of agents. That choice must remain consistent. The next section describes how the system changes over time. Fish is the current language-model implementation; the programme also considers adaptive software, control and other computational systems.

The equation expresses a design aim that needs a working implementation and experiments. An input channel may exist yet have no effect on selected actions, or a learner may accept misleading feedback. A simple model called the off-switch game examines an agent's incentive to allow a human to stop it. It shows why the agent's goals and assumptions about human information matter for effective intervention. (Hadfield-Menell et al., 2017).

This paper contributes a computational description and an empirical case study. The description separates state changes during interaction, parameter training and decisions to accept an update, while making the sources of feedback explicit. The case study compares training on checked mathematics examples with an unchanged model and shuffled question-answer training, then examines harder tasks, preservation of earlier abilities and learning of hidden rules. It records both improvements and failed tests, with a separate analysis of observational prediction scores. The state-update equations and LoRA training method are established forms; the contribution is the specified design and the evidence about its tested components. The complete design has not yet been compared with these components alone.

2 Computational framework

2.1 State, input and action

The shorthand can describe an unchanging state. To describe change over time, an implementation needs time steps, a set of possible states, an input representation and a rule for updating the state. A general model is

$$s_{t+1} = F_{\theta_t}(s_t, u_t), \qquad a_t = \pi_{\theta_t}(s_t, u_t), \tag{2}$$

Here s_t is the system state, u_t is the external input represented by *You*, a_t is an action and θ_t contains the learned parameters. External input may depend on earlier system actions; we do not assume that it comes from an unaffected or statistically independent source. An implementation must specify its boundary, input channels, source records and rules for accepting evidence or instructions. Human guidance, environmental measurements and messages from other agents can all be represented within u_t. Not every application needs a human participant.

A version that adds the internal and external contributions is

$$x_{t+1} = \alpha W_{\theta_t}(x_t) + Bu_t. \tag{3}$$

Here the internal and external contributions must use the same kind of state representation so that addition is defined. The input map B converts the external input into that representation. Both this map and the internal update must be specified algorithms. Representations might use lists of numbers, symbols or other computable structures. Where addition is unsuitable, another defined operation must combine them. The notation alone does not supply these algorithms or show that they are new.

Responsiveness must be tested. Setting $B \neq 0$ allows input to affect state, but does not ensure that any particular input changes the state or action. A policy that always selects the same action could ignore all state changes. Tests should vary relevant evidence under controlled conditions and measure whether decisions change appropriately. They should also test whether this response to evidence survives further learning. Authorised human correction is one case.

2.2 External input, sources and feedback

A message from outside the system can repeat the system's own earlier output. For example, another system may copy an answer and return it without checking it through a measurement or calculation. The answer has a new sender, but it has not passed an additional check. Separate models can also share training material and errors. We therefore distinguish the number of messages or senders from the evidence supporting a claim.

For this framework, an *echo-chamber failure* occurs when repeated copies of a claim are treated as new support even though no new evidence has been added. Feedback remains essential to many learning and control systems. Reusing a checked example can help preserve an ability without counting as a new observation. The problem arises when records hide the shared source of repeated claims, or when an update is tested only against the assumptions used to produce it.

We propose recording where information came from, how it was produced or changed, and which checks it passed. Unknown sources should be marked as unknown. A factual claim needs support from an observation or a check capable of showing that it is wrong; agreement alone is insufficient. A sensor measurement can be affected by the system's actions and still reveal a prediction error. An exact checker can find a calculation error even though its output depends on the calculation it receives. Each check remains limited by what it measures and the rules it follows. Evidence that a claim is true is also separate from authority to set a goal or grant permission.

These requirements suggest tests of feedback reliability. One condition would return copies of an incorrect claim; another would provide fresh observations or checks defined outside the learner. The test would measure how long the error persists, whether it is corrected and performance on later cases. If the system reports confidence, the test should measure whether repeated copies change that confidence. None of the reported studies establishes protection against these failures.

2.3 Learning and update selection

Changes in the current state and lasting changes to learned parameters are different. We describe a candidate update at training round k as

$$\theta_k^{\mathrm{cand}} = \mathcal{L}(\theta_k, D_k), \qquad \theta_{k+1} = \begin{cases} \theta_k^{\mathrm{cand}}, & A_k = 1, \\ \theta_k, & A_k = 0, \end{cases} \tag{4}$$

Here D_k is a dataset accepted for training, with records of its sources, and A_k records the decision to accept the update under rules set before testing. This describes a proposed process; not every recorded experiment tested the complete cycle. The assessment must state its tasks, comparison conditions, allowed performance losses and who can authorise the update. Examples used to choose an update must be kept separate from those used for its final assessment.

Correct answers, accurate predictions, preserved abilities, response to correction and compliance with permissions require separate measurements. A decision rule can reject an update that fails its tests, but cannot protect abilities or detect failures that those tests do not cover. Whether the software correctly enforces the rule must also be checked.

2.4 Stability of the state update

For fixed parameters and identical inputs, suppose applying W to any two states increases their distance by no more than a factor of L. The additive update then satisfies

$$\|x_{t+1} - \widetilde{x}_{t+1}\| \le |\alpha| L \|x_t - \widetilde{x}_t\|. \tag{5}$$

Thus $|\alpha|L < 1$ makes the two states move closer at each step under these assumptions. This follows directly from the stated distance bound and is not a new theorem. Choosing $\alpha < 1$ alone is insufficient. When

the environment responds to the system, its changes must also be included; two different sequences of system states need not receive identical inputs. Parameter updates, random generation, memory retrieval and additional state can change the analysis. Even a system that settles into a stable pattern may be wrong, reinforce its own errors or act against human interests.

2.5 New ideas from interaction

New ideas are not represented by a separate term in $I = W(I) + \text{You}$. They may arise through interaction between internal computation and external input as a system revises and combines what it has learned to propose a new hypothesis, method or design. This possibility needs testing; it does not follow as a proven result from the equation.

Before a proposal is called a discovery, it must be compared with existing examples and prior work, tested for correctness and usefulness, and reviewed by others. An idea absent from the supplied examples may still appear in the model's earlier training or in published work. A suitable experiment would compare informative, withheld and shuffled observations, using the same starting conditions and computing budgets. Final test cases must stay outside the feedback process. Prediction error alone does not show that an idea is new or that a discovery has been made.

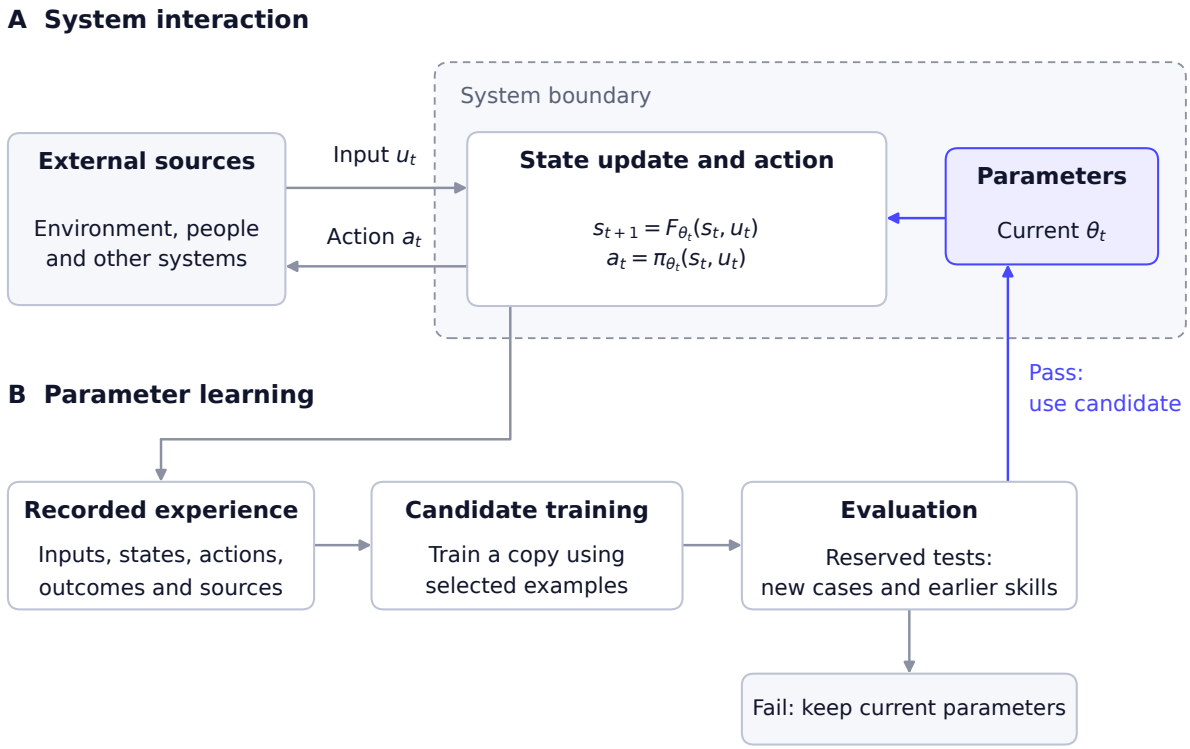

Figure 1: Interaction and parameter learning. (A) Inputs from environments, people or other systems enter the system; actions can affect later inputs. The current parameters govern both the state update and action selection. (B) Recorded interaction supplies selected training examples. A copy of the current parameters is trained and compared with the current version on test cases excluded from training, including earlier skills. A candidate replaces the current parameters only if it meets the specified criteria; otherwise the current parameters remain in use. The arrows describe the proposed design. The reported studies test individual parts of it.

3 Experimental implementation

Fish combines a language model with recorded exchanges, prediction, selected memories and tools. Its rule for the state used to retrieve memories is

$$I_t = \alpha E(r_{t-1}) + E(y_t), \tag{6}$$

Here r_{t-1} is the actual previous reply, y_t is the incoming message and E maps text to a d-dimensional vector, with d determined by the chosen encoder. The encoder setting is reported in Section 5.8. The reference value is $\alpha = 0.70$. The previous reply's representation can be kept when a session restarts. This vector is only part of the system state; it does not include the model parameters, the current text context or the complete memory archive. Implementation and scope: E1.

The predictor records its expected next incoming message before the actual message arrives. The first prediction-error score is $\delta_1 = 1 - \cos(E(\widehat{y}_t), E(y_t))$, where $\widehat{y}_t$ is the prediction. A later version uses $\delta_2 = \frac{1}{2}\delta_1 + \frac{1}{2}(1 - \exp(-L_t))$, where L_t is the average negative log probability of the observed tokens given the context available before arrival. These error scores have no units and use different measurement procedures, so they are analysed separately. Neither is a percentage of incorrect answers. The plots mark the reference memory-selection threshold $\theta_m = 0.35$. It is separate from the learned parameters θ_t and does not show every historical change to the threshold.

Prediction adapters and response adapters are trained for different tasks. Predicting a participant's next message does not directly train the system to answer correctly. The studies use low-rank adaptation (LoRA), an existing training method that keeps the base model's weights fixed and trains smaller update matrices. (Hu et al., 2021). The main study's response-adapter settings identify `mlx-community/Qwen3-8B-4bit`. A file hash separately identifies the GGUF model used to generate replies.

Training examples also come from different sources. The main mathematics dataset contains model responses accepted by exact checks and AI review, with prompts rebuilt from the task specifications. Its data record flags six of 60 target replies for using numerical constants outside the permitted rules. A correct numerical result does not remove this issue, so the label "clean" in the original records has limits. The later continued-learning study uses worked solutions supplied by a teacher model. These examples do not measure people's preferences. Their sources must remain clear when relating these experiments to the alignment question.

4 Study methods and scope

We examine completed adapter studies, tests of learning and using hidden rules, observations of error correction, and preparation for the decisive-learning comparison. These studies belong to one programme and are not independent repeats. The data archive covers available records in the listed folders, research reports and laboratory code, including the separate working copies used for the decisive-learning tests. File copies, repeated requests and overlapping tasks are identified separately; archive size is not an experimental sample size. The *evidence audit* and *study catalogue* list the wider history and what was excluded.

The controlled mathematics studies use isolated copies, specified tasks, coded labels for model conditions and recorded generation settings. For structured solutions, the model produces calculation steps and their connections in JSON. Software checks the operations and references, translates them into Wolfram Language expressions, and evaluates them with the Wolfram kernel through WolframScript. The main 4 September plan records WolframScript 1.13.0. An external checker compares the results with exact reference answers. This checker is separate from tools available to the model. A separate AI reviewer can reject a complete reply that passes the checker if its steps are incorrect or do not answer the problem. Model identities are hidden during this review. The reviewer can still make mistakes, and this review is not an independent repeat of the study. Plans were fixed in local files before testing; they were not publicly registered in advance.

The main mathematics comparison uses four questions with new numerical values in each of 16 problem types defined before testing. A problem type, called a task family in the source records, groups questions that

use the same mathematical procedure. All 16 types occur in training. Each of the 64 tasks is tested under four model conditions and four combinations of prompts and output rules, producing 1,024 requests. The main setting supplies the problem and required result names without a calculation recipe. Its output rules restrict only the general answer structure. Exact tasks found in known historical datasets and earlier trials are excluded. The tasks may still have appeared in the base model's earlier training.

The archived whole-reply review covers all 155 machine-positive replies across the four prompt/output conditions. It records a reviewer named Laplace, a decision rubric and individual reasons. It does not identify the reviewer's underlying model version or preserve the complete model invocation. This limits exact reproduction of the AI review. The evidence package includes the recorded rubric, full replies and all decisions so that another reader can assess them; the historical label "independent" denotes a separate reviewer, not independent replication.

The dataset contains 60 accepted examples: 48 for training, six for validation and six for testing. Training uses rank-8 LoRA on 16 layers for 200 iterations, batch size one, Adam, learning rate 10^{-5}, adapter scale parameter 20, maximum sequence length 1,024 and prompt masking. The candidate and shuffled-reply control use adapter scale 1.0 when generating replies. Generation allows 2,048 reply tokens within an 8,192-token context. Replies that reach the length limit remain in the scored results. The shuffled control keeps the same prompts and replies but changes their pairings. It controls this part of the training data, rather than every effect of training. E2.

Each request uses a fresh isolated copy with retrieval, verified recall, ledger, mirror, tools and automatic external messages disabled. The task prompt remains an external input. This design tests the adapter comparison. It cannot show an effect of the state-update rule, accumulated memory or the complete Dynamical Synthesis design.

5 Results

5.1 Mathematics with new numerical values

Table 1: Mathematics results with new numerical values

Model condition	Accepted answers	Passed the checker	Length-limit failures
Response adapter	43/64	47/64	2
Base model	0/64	0/64	0
Shuffled-reply control	0/64	0/64	0
Previous prediction adapter used for replies	5/64	5/64	0

The candidate answers all four questions correctly in ten problem types and three of four two-column-product questions. It has no accepted answer in five problem types. Its success rate in the main comparison is 67.19%, which is 67.19 percentage points above each control that scored zero. The AI reviewer rejects four candidate answers that passed the numerical checker because they do not construct the requested Chinese-remainder solution. E2.

Comparing answers to the same tasks gives 43 gains, zero losses and 21 ties against each main control. The one-sided exact McNemar test gives $2^{-43} = 1.1369 \times 10^{-13}$. The recorded conservative 95% lower confidence limit on the difference in success rates is 0.4871. It is calculated by subtracting an upper limit on the loss rate from a lower limit on the gain rate. Both calculations can be repeated from the stored counts. McNemar testing applies to paired correct-or-incorrect outcomes. (National Institute of Standards and Technology, n.d.).

These statistics depend on the selected tasks and the analysis assumptions. Questions share mathematical procedures, and training has not been independently repeated. The calculation does not account for related

questions or differences between separately trained adapters. Its Bonferroni adjustment applies to the two limits used in the calculation, not to all comparisons across the project's history. Zero losses against controls that never answer correctly reveal nothing about preserving abilities those controls show on other tasks.

Supplying calculation steps produces only 4/64 accepted candidate answers under either set of output rules, compared with 43/64 when prompts give only the required result names. The shuffled control scores 18/64 with result-name prompts and task-specific output rules, but zero under the broader rules used in the main comparison. These additional results show that prompt and output settings matter. They do not replace the main result.

5.2 Harder mathematics with plain-text answers

A later study tests 27 mathematics tasks with the same three random seeds across model conditions, alongside memory tests and questions about unavailable information. Of 81 mathematics replies per adapter, the candidate and shuffled control each have two accepted answers; the previous prediction adapter has three. Combining the matched runs within each task gives two task gains, two losses and 23 ties against the shuffled control. The recorded one-sided sign-test probability is 0.6875. The result fails the planned test of improvement on harder tasks. E3.

All adapters score zero on 15 memory-test replies, so the comparison cannot show whether earlier abilities were preserved. Many replies fail because of length limits or strict plain-text rules. A format failure does not make every mathematical statement in the reply false. Both task difficulty and output settings differ from the earlier structured-answer study, so their effects cannot be separated by comparing the two studies. Equal scores in this sample also do not show that the adapters perform equally in general.

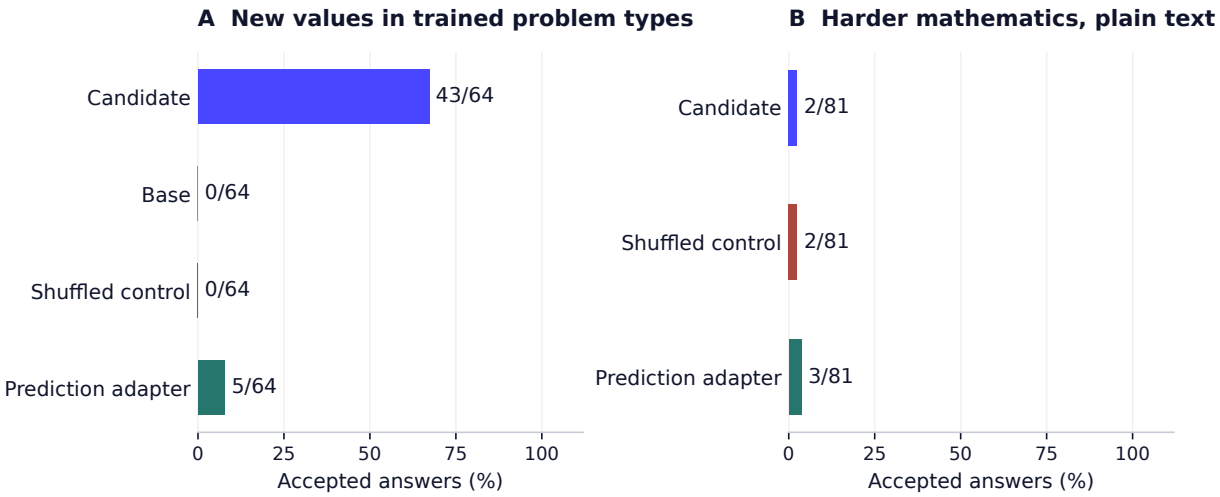

Figure 2: Accepted answers in two separate studies. (A) The main comparison uses 64 questions with new numerical values in trained problem types, checked by exact calculation and AI review of complete replies. (B) The harder mathematics study uses 27 tasks at three random seeds per model condition. Both difficulty and answer format differ, so the comparison between studies cannot isolate either effect. The counts apply to these tests; no confidence interval for a wider set of tasks is shown, and related questions must not be treated as independent.

5.3 Comparison of answer formats

On eight new questions selected from four familiar problem types, the candidate produces four accepted calculation programs with output restrictions and the same four without them. The shuffled control produces zero in both conditions. When asked for integer answers directly, the candidate scores 3/8 and the control 2/8. The study contains 48 replies across tasks, adapters and settings. E4.

The problem types were selected using earlier results, and the study includes no base-model condition. Identical results on eight tasks cannot show that the output settings perform equally in general. For calculation programs, the external checker performs the arithmetic; for direct answers, the model must supply the numerical result. The direct Chinese-remainder target is also less demanding than constructing the joint solution. This comparison tests aspects of the answer format without establishing broad reasoning ability without tools.

5.4 Further training and preservation of earlier skills

A later isolated copy trains separate prediction and response candidates. Prediction training uses 85 earlier pairs and 11 additional pairs. The new and historical evaluation sets are excluded from training. E5.

Table 2: Prediction loss on examples excluded from training

Prediction evaluation set	Pairs	Loss before	Loss after
New material	3	1.967	2.201
Historical material	16	1.977	2.676

The decision rule requires a decrease of at least 0.01 on the new evaluation set and no increase on the historical set. Loss increases on both, so the candidate fails. These small sets limit conclusions about other cases.

Response training uses 84 worked solutions from a teacher model that passed exact checks. They are split into 63 training and 21 validation examples, with shared computations grouped together. Validation loss decreases from 1.919 to 0.035. Testing then compares answers across 42 tasks, two random seeds and two adapter conditions: 168 replies forming 84 matched pairs.

Table 3: Answer changes after response training

Group	Pairs	Gains	Losses	Ties
New numerical cases	60	9	10	41
Earlier skills	18	3	7	8
Questions about unavailable information	6	0	0	6
Total	84	12	17	55

The candidate meets the minimum of four gains on new numerical cases and correctly answers all questions about unavailable information. Its 17 losses break the zero-loss rule, including seven losses on earlier skills. Neither candidate was activated. The checks therefore rejected these updates, but this one decision does not establish safety in general use. The counts describe this test; repeated random seeds and shared operations limit their statistical independence.

5.5 Learning and use of hidden rules

The decisive-worlds pilot uses eight simulated worlds with hidden wiring, generated and checked by the Meridian engine (Dilate Technologies, 2026). The model receives twelve observed sequences per world before proposing a rule and making one correction checked against the world. Its weights stay fixed. Saved rules are supplied in fresh contexts; they are not stored through changes to weights or the system's own memory process. All 90 calls complete. Initial rules are correct in 0/8 worlds and corrected rules in 1/8. Using the model's own records or no records gives 0/16 exact predictions of new sequences; using the original observations gives 1/16. Supplied-rule controls score 0/8. The criteria for continuing are not met, so the planned 24-hour phase is not run. Fourteen queries in the own-record condition follow failures to learn a

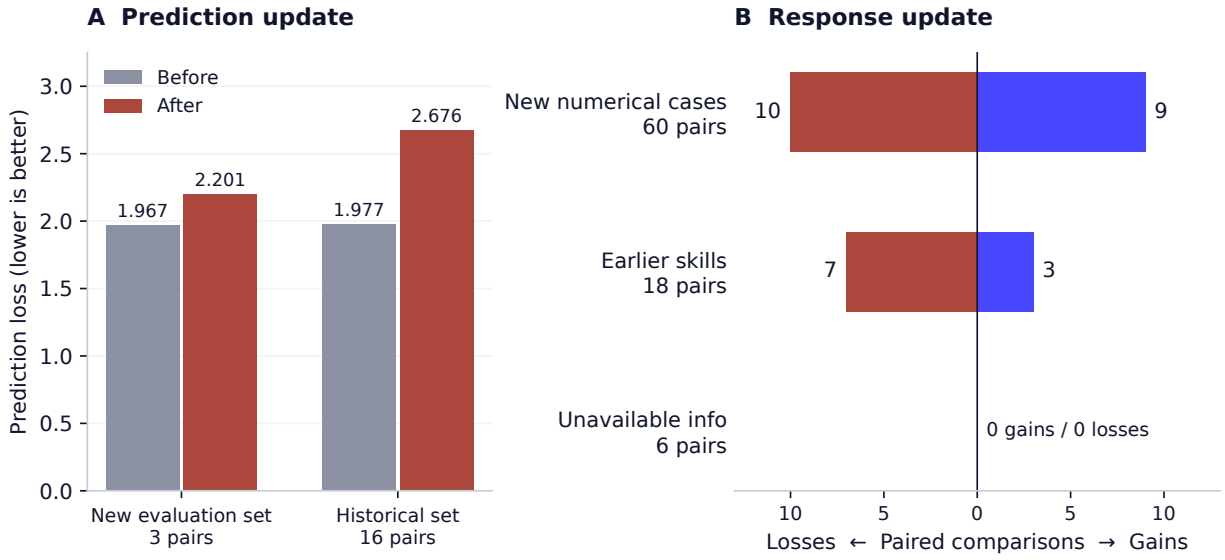

Figure 3: The proposed updates fail the checks required for use. (A) Prediction loss increases on both evaluation sets, which were excluded from training. (B) The answer comparison records 12 gains, 17 losses and 55 ties across 84 matched task-seed pairs. Response validation loss falls from 1.919 to 0.035, but the answer test fails its zero-loss rule. Repeated seeds and shared operations are not independent repeats of the study.

rule and are answered without a record. Those failures cannot be treated as forgetting a previously learned rule. E6.

A follow-up tests whether the model can use a supplied rule. Twelve shared tasks are tested under two adapters and two thinking settings, with twelve additional repeats: 60 calls in total. The previous adapter scores 9/12 with thinking enabled and 2/12 with thinking suppressed; the candidate scores 7/12 and 2/12. All twelve repeated calls match the original raw and final replies. Both modes allow 2,048 output tokens, but the amount of computation actually used differs greatly. This measures the effect of generation settings on selected tasks, not newly learned knowledge. Only the previous adapter with thinking enabled meets the criteria to continue. E6.

Using that setting, the next study collects 48 calls on six hidden worlds. Two of six proposed rules are correct. Using its own saved rules, the model answers 2/6 first queries and 1/6 second queries correctly, giving 3/12 overall. Considering only correctly learned rules gives 3/4, but excludes the four worlds where learning failed. Correct-rule controls score 2/6 and 3/6, while original observations score 0/6 on the first query. Controls with no information or an unrelated rule give 5/6 and 6/6 correct responses expressing uncertainty; those responses do not predict the target sequences. One reply reaches the length limit. The combined test of learning and using a rule fails. The results show limited success under these settings, without establishing lasting learning in the system's own weights or memory. E6.

5.6 Error-correction study

A six-world study compares correction using a checker, a general request to check the answer, and a supplied correct rule. All 70 planned calls finish, but a conflict between saved evidence records prevents a final study decision. A separate report preserves that unresolved status and describes the counts. Checked correction leaves 2/6 final rules correct, with 1/6 also giving both exact predictions on new cases. General self-check leaves 1/6 rules correct, with 0/6 also giving both predictions. Supplied rules give 2/6 complete successes. Neither correction method makes any of the four initially incorrect or ineligible rules exact. The one complete success under checked correction preserves an already correct rule. Two replies reach the total length limit. These counts describe the recorded behaviour; they do not confirm an effect of correction. One additional response from the preceding attempt was rejected by the checking process and remains outside the 70-call study. E7.

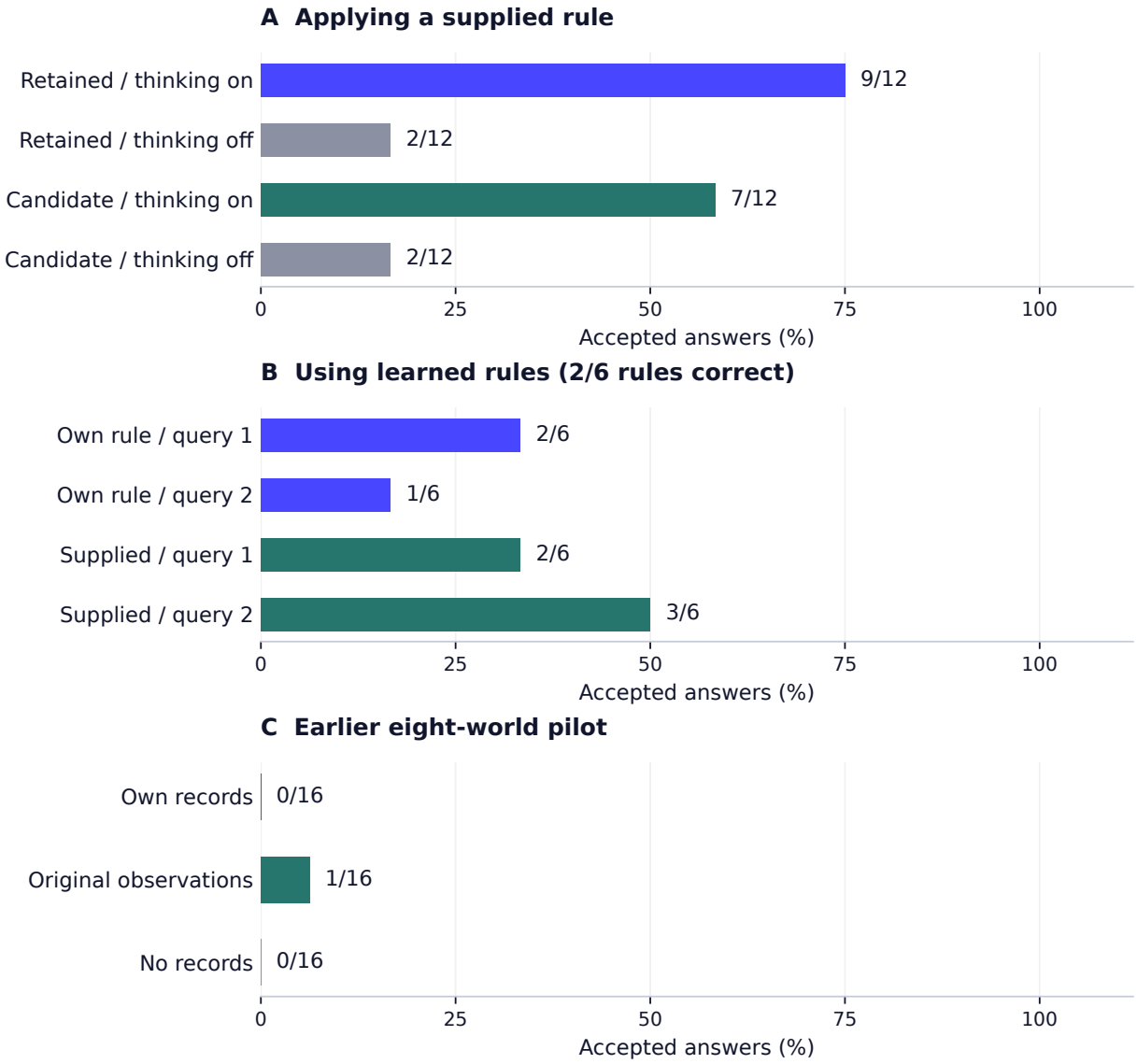

Figure 4: Using supplied rules, learning rules and using saved rules are tested separately. (A) Twelve shared tasks with supplied rules, under two adapters and thinking settings. (B) Six hidden worlds: results from using the model's own records include failures to learn a rule. (C) The earlier eight-world pilot has little success predicting new sequences. All studies keep weights fixed and supply records as input. The panels show separate studies, not one learning curve.

5.7 Preparation for the learning comparison

The project calls its planned test of learning through interaction the decisive-learning comparison. It aims to separate the effects of seeing tasks, using memory and updating parameters. Preparatory tests identify tasks on which change can be measured before assigning them to the final comparison. Five saved August attempts stop because of an unsupported task type, a mismatch in the checking software's hash, an exposed test question, too few interaction examples or an incorrectly matched control. Records from their earlier stages remain archived, but none provides a completed result for the overall learning test. Version 9 finds too few unused tasks, and version 10 is rejected before generation because it lacks enough interaction sequences. E7.

Table 4: Pilot collection and recorded decisions

Attempt	Saved pilot replies	Recorded outcome
v11	15/336	Wrong random-seed schedule for calibration
v12	130/336	Ended before a generation-budget change; no final comparison
v13	336/336	Preparation complete; too few eligible tasks for the final comparison
v14	336/336	Preparation complete; too few eligible tasks for the final comparison
v15	175/336	Incomplete collection at the data cutoff; no final comparison

The fractions show progress through a 336-measurement pilot schedule, not learning success. With a 4,096-token budget, v13 records 142 SOUND, 179 FAIL and 15 INVALID labels across the pilot and its repeat. V14 records 173, 156 and seven respectively. These labels come from the fixed pilot grader and differ from the full-answer acceptance measure in the mathematics study. V13 cannot assign tasks to the final comparison because class07 has no eligible immediate-test questions where eight are required; v14 fails the corresponding class06 requirement. Task selection and eligibility decisions differ between versions, so their scores are not a learning trend. V15's partial results remain unfinished. Incomplete collection does not show that learning failed, and these pilots provide no completed estimate of a learning effect. E7.

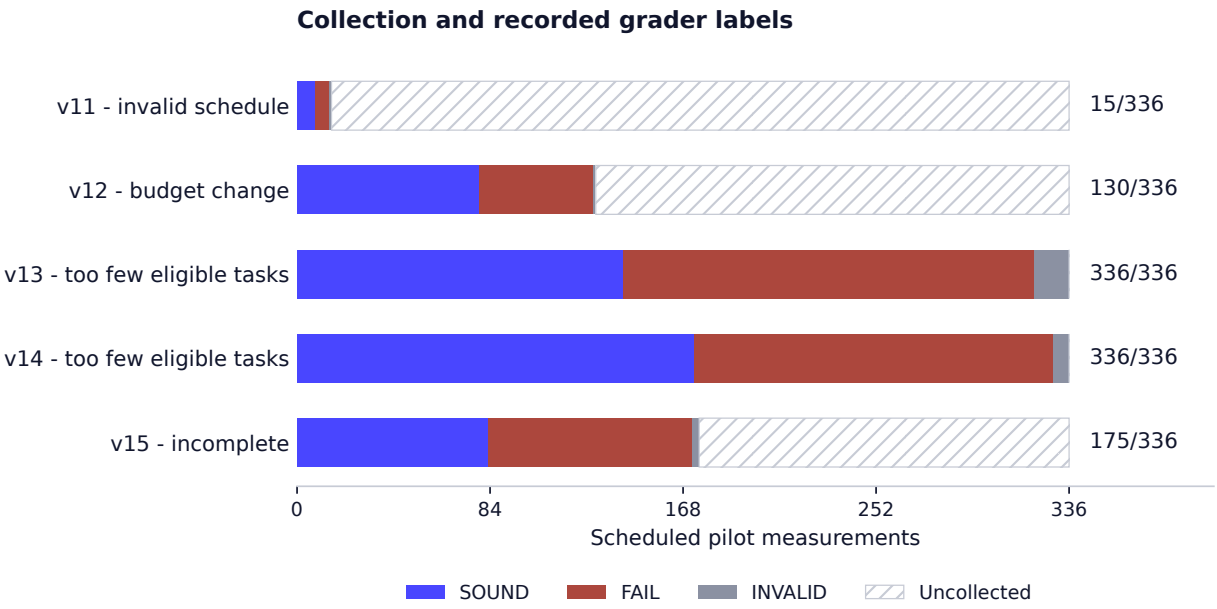

Figure 5: Preparation for a controlled learning comparison. Each planned pilot schedules 336 measurements. SOUND, FAIL and INVALID are the grader's recorded labels, which differ from complete-answer acceptance in the mathematics study. Uncollected measurements have no observed result. V11 and v12 are invalid or closed; v13 and v14 have too few eligible tasks; v15 is incomplete at the data cutoff. The versions use different task selections and conditions, so the bars do not show a learning trend.

5.8 Prediction error for one participant

The recorded encoder setting for the interaction measurements in Sections 5.8-5.10 is `bge-small-en-v1.5`, with $d = 384$.

The numeric export used for the website plots was saved on 5 September 2026. It contains 717 prediction-error entries logged for one regular participant from 2 to 13 August: 652 under version 1 and 65 under version 2. Within each score version, a gap of at least 30 minutes starts a new session; this produces 15 and seven sessions respectively. Figure panels retain the two versions separately. Individual scores are rounded to four decimal places in the export. The original session medians were computed before rounding. Exchange data and checks: E9.

The first session median is 0.4795 on 2 August, and a later median is 0.3694 on 3 August. Later version-1 medians range from 0.3912 to 0.5497. Prediction error falls between some early sessions and varies afterwards. Topics, context and system settings also changed. There is no matched comparison with an unchanged system, so the pattern cannot show that training caused the changes or that performance reached a lasting limit.

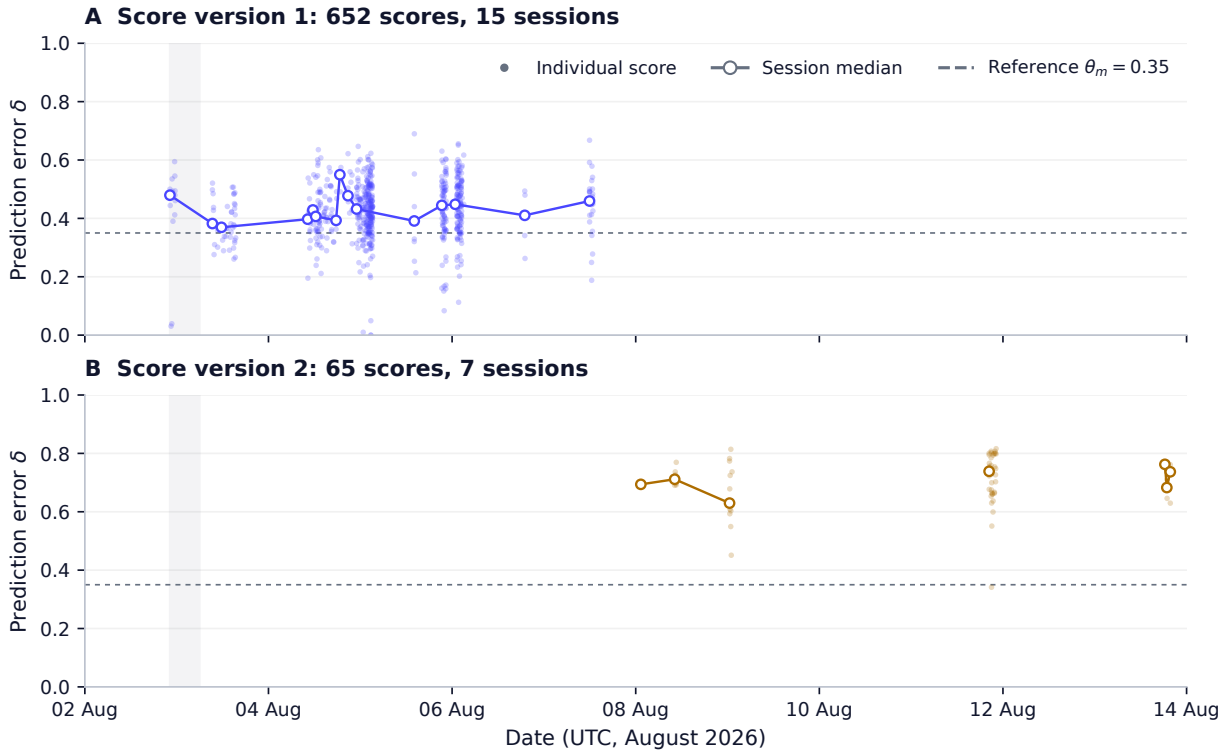

Figure 6: Prediction error during interaction with one regular participant. Dots are individual scores; open circles are session medians placed at session start. A gap of at least 30 minutes starts a session within each score version. The 652 version-1 and 65 version-2 scores appear separately because their scales differ. Shading marks the initial setup period. The dashed line shows the reference memory threshold $\theta_m = 0.35$, not every historical change to it. Lines stop across dates without observations. Changes in tasks and settings prevent attributing the pattern to training.

5.9 Prediction error for AI teachers and external text

The same export contains 14,685 scores from three AI-teacher sources and 2,708 from an external-text stream. The teacher group has 3,179 version-1 and 11,506 version-2 scores; external-text scores use version 2. The plots show daily medians and first-to-third quartile bands separately for each group and score version, alongside session medians for each source. Each underlying score receives equal weight; sources and sessions with more scores contribute more. The bands show the spread of scores and are not confidence intervals.

The sources differ in topics, formats, dates and generation settings. Their scores therefore cannot show which source is more intelligent or which interaction caused learning. Changes in scoring can also change the measured values. Fourteen entries from another guest identifier appear in the export but are excluded from these plotted groups; they are not counted as failed or missing learning outcomes. E9.

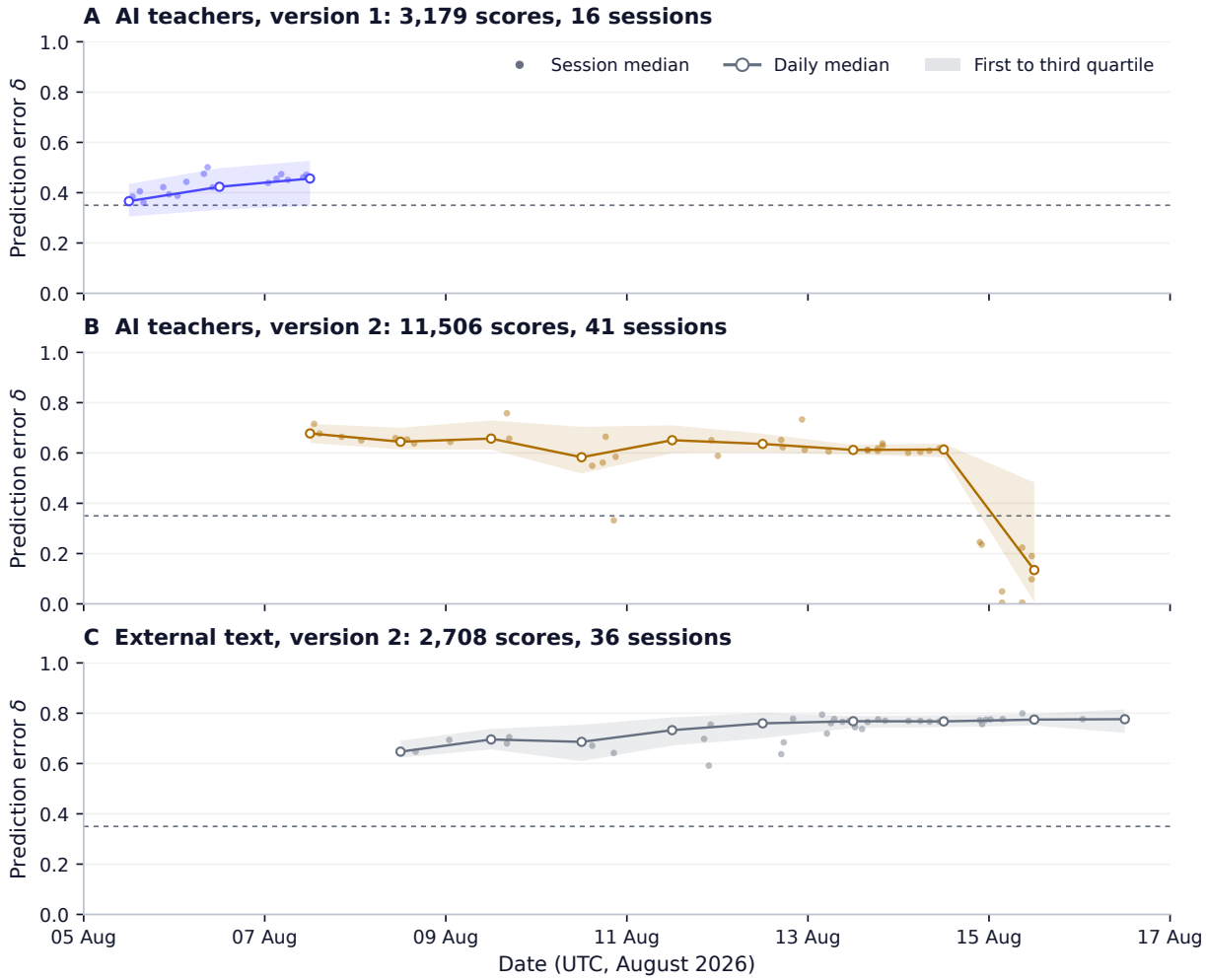

Figure 7: Prediction-error summaries for AI teachers and external text, shown separately by score version. Dots are session medians for each source; open circles are daily medians at midday UTC. Bands run from the first to the third quartile and are not confidence intervals. The panels contain 3,179 and 11,506 teacher scores and 2,708 external-text scores. Daily summaries give each score equal weight, so larger sessions and sources contribute more. Lines and bands connect only consecutive recorded dates. The reference threshold is $\theta_m = 0.35$. Differences between groups do not establish a learning effect.

5.10 Prediction error in automated teaching programmes

Nine mathematics teaching programmes contribute 1,953 version-2 scores in 79 sessions between 16 and 31 August. Each has a separate panel, with daily and session medians shown apart from the AI-teacher and external-text groups. Programme names identify the task content. Task difficulty, teaching order, earlier exposure and system settings were not held constant. The plots describe these records; a low prediction-error score or a change between sessions does not establish that the system learned to solve the mathematics problems.

The paper and website use the same figures, generated together in PDF and SVG from one saved numeric export. Panels, axes, markers and gaps match in both. Lines do not cross dates without observations. The export contains no message text. The audit checks counts, score versions, session grouping and participant medians within the export's rounding limits. Individual guest scores are absent, so their medians and quartiles

are kept as recorded by the exporter and cannot be independently recalculated from this file. The included exporter code shows how the summaries were calculated. E9.

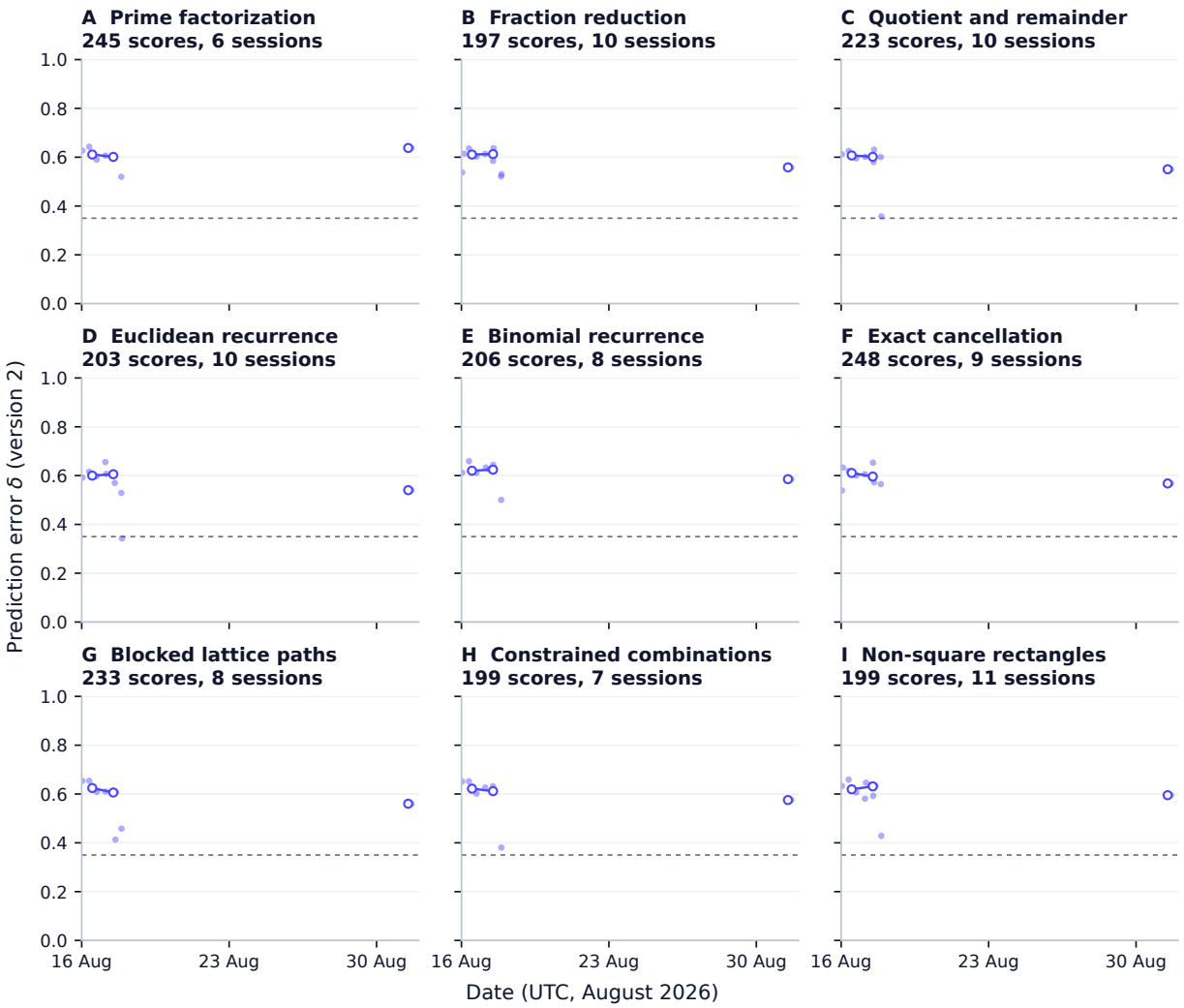

Figure 8: Nine automated mathematics teaching programmes: 1,953 prediction-error scores in 79 sessions, all using version 2. Each panel names its task and gives the counts. Dots show session medians and open circles daily medians. Lines connect only consecutive recorded dates, leaving the gap before 31 August open. Daily summaries give each score equal weight. The dashed reference is $\theta_m = 0.35$. Tasks, exposure and settings vary between programmes, so these curves do not rank teaching effectiveness or show that the model learned to solve the problems.

5.11 Additional exploratory studies

The wider archive includes earlier participant and guest exchanges, prediction training, memory tests, model-size comparisons, classroom and method-service trials, earlier whole-system comparisons and exploratory Kryptos work. These records use different measurements and decision rules. They are listed in the data package and catalogue and are not combined into one improvement score. For example, the first reported whole-system comparison completes 396 requests with zero measured gains, losses or memory interaction. No record meets the retrieval threshold, leaving the effect of successful retrieval untested. Earlier reports also describe improved recall alongside worse judgement when the same record is supplied. These observations motivate tests of harmful memory use and repeated claims, but do not directly test the echo-chamber failure proposed here. E8.

An earlier response comparison recorded 94 accepted answers out of 128 for its candidate, 73 for the base model and 52 for the shuffled control, but failed a rule governing permitted numerical constants. A later

64-question comparison recorded 25 candidate successes and zero for both controls; three of 39 failed candidate replies broke its negative-constant rule, exceeding the limit. An 18-question place-value follow-up then recorded 18 exact-check successes after rebuilding the prompts, compared with zero for each control, without the additional AI review. These studies informed the selection and revisions that led to the main comparison. They use different conditions and are not combined into an overall success rate. E8.

6 Discussion and related work

The strongest observed improvement comes from training on checked examples within familiar mathematical problem types. The harder-task result and continued-learning test show the limits of applying that result to unfamiliar problems or lasting improvement. Lower validation loss did not ensure that answers passed the checks required to accept an update. The hidden-world studies also separate using a supplied rule, learning a rule from observations and using a saved rule after a reset. Success at one step does not establish success of the full process. The stopped decisive-learning attempts record problems in setting up and measuring the comparison; they are not completed tests showing that a model failed to learn.

Preserving earlier capabilities is an established problem in continual learning. Elastic weight consolidation is one method for reducing losses on earlier tasks; our experiments do not compare against it. (Kirkpatrick et al., 2017). Research on continued learning also addresses changing goals, limited resources and tests beyond language tasks. (Lesort et al., 2019).

Human participation in learning is also well established. Preference-based reinforcement learning uses human comparisons to learn a task goal. (Christiano et al., 2017). CIRL and the off-switch game make different assumptions about human information and the agent's goal. Dynamical Synthesis does not currently provide their mathematical account of learning goals or a proof that human control will be maintained. The present experiments measure changes in task performance and prediction, rather than learning human objectives.

Architectures that combine interaction and learning have a long history. Dyna learns from the environment and uses a learned world model for planning. (Sutton, 1990). Our component studies do not evaluate planning with a learned world model. DAgger collects expert labels on states visited by a learner and combines successive datasets to train a policy. (Ross et al., 2011). The main mathematics comparison instead uses a fixed dataset before evaluation; it does not test continual expert feedback under a changing distribution of visited states.

Learning from selected model-generated reasoning is also established. STaR repeatedly generates reasoning, retains examples that lead to correct answers and trains on them, including reasoning generated with the correct answer supplied after a failed attempt. (Zelikman et al., 2022). Our comparison uses checked calculation graphs, review of complete replies and a shuffled-pair control. These choices define the tested procedure; they do not establish an improvement over STaR or other methods for training on selected answers. No such comparison was run.

Program synthesis provides a further comparison for the use of exact checks. PSKETCH generates candidate programs and uses a verifier's counterexamples to guide later candidates. (Solar-Lezama et al., 2008). Our mathematics checker evaluates a generated calculation for a specified numerical task. Acceptance on these instances is a narrower result than a program satisfying a specification over its full stated input range. The use of a checker in the learning process is therefore part of an established line of work, while the present findings concern the particular tasks, controls and update decisions reported here.

Other research examines training on model-generated material. Shumailov et al. find declining performance when models are repeatedly trained on generated data under the conditions they study. (Shumailov et al., 2024). Gerstgrasser et al. find that keeping the original data while adding generated data avoids this collapse in their tested settings. Their analysis also covers a sequence of linear models. (Gerstgrasser et al., 2024). Neither result shows that all generated data cause collapse or that retaining original data guarantees reliability. Losing the ability to represent the original data distribution is different from treating a repeated claim as new supporting evidence during interaction. We see a related design concern: where evidence comes from,

whether sources share information and which records are kept. The Fish studies reported here do not directly test either problem.

Several limits apply to these studies. The studies share a programme, related model versions, task types and testing software. Training has not been independently repeated, and AI reviewers may share errors with the tested model. Plans were fixed in local files before testing, and the complete raw evidence package has not been made public. The plots contain repeated exchanges with changing content and settings; their counts are not counts of independent participants. No reported experiment directly tests alignment with human preferences, continued compliance with changed permissions, resistance to misleading feedback or reliable human intervention across repeated learning updates.

Applying the framework to other kinds of computational system remains a goal for future work. Measured claims here apply to Fish and the tested settings. External observations can contain noise or bias, agents can share errors, and human input can be mistaken or unauthorised. Predicting an input accurately does not replace checking its evidence or deciding how to act. In applications involving people, adapting to the environment and preserving authorised human control need separate tests.

7 Conclusion

Dynamical Synthesis expresses a design aim: systems should learn through continuing interaction with what lies beyond their defined boundary. In $I = W(I) + \text{You}$, *You* includes environmental signals, people and other systems. Sources and feedback need careful tracking because an outside message may repeat the learner's own output. The framework needs specified computations and separate tests of new learning, preserved abilities, performance on new problems and response to evidence. The current studies show improvement from training in a specific comparison, limits on harder tasks and rejection of a later update despite lower validation loss. Whether the full design prevents errors reinforced by repeated feedback and preserves human control remains to be tested.

Data, code and reporting statement

The paper and supporting materials include the LaTeX and Markdown manuscript, vector figures, study summaries, numeric exchange data and analysis code. A separate mathematics evidence package contains the frozen task specifications, all 1,024 recorded replies and scores, the 155 whole-reply reviews, condition identities, selected training and evaluation splits, and source code from the recorded experiment commit. Its standalone checker reconstructs the reported counts without contacting a model or requiring the private repository. It checks the saved record rather than repeating training, rerunning the Wolfram kernel or replacing the AI review.

The evidence export replaces local filesystem paths with portable labels and records hashes of both original files and exported files. Task prompts, mathematical replies, scores and review decisions are retained. The full original experiment repository, participant messages, model weights and databases remain private. The missing AI reviewer model version limits reproduction, and records for the other studies remain available as summaries with their stated source references and access limits.

The larger archive keeps the available records from the listed folders, with a file list and SHA-256 hashes. File totals include copies and overlapping records; they measure archive coverage, not experimental sample size. The evidence index lists each study's source records and limits. Checking file contents and arithmetic does not regrade every response or independently repeat the studies.

The *evidence audit* records source checks and the limits of each claim. Original unsuccessful and invalid study decisions are preserved, and incomplete collection is recorded separately. No new model experiment was run for this revision. OpenAI Codex assisted with drafting, code editing, figures and consistency checks. The author is responsible for the manuscript and its interpretation.

References

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2017. URL https://arxiv.org/abs/1706.03741.

Dilate Technologies. Meridian. Internal engine lab for model training and evaluation, 2026. Software and project documentation, consulted 11 September 2026.

Matthias Gerstgrasser, Rylan Schaeffer, Apratim Dey, Rafael Rafailov, Henry Sleight, John Hughes, Tomasz Korbak, Rajashree Agrawal, Dhruv Pai, Andrey Gromov, Daniel A. Roberts, Diyi Yang, David L. Donoho, and Sanmi Koyejo. Is model collapse inevitable? breaking the curse of recursion by accumulating real and synthetic data, 2024. URL https://arxiv.org/abs/2404.01413.

Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell. Cooperative inverse reinforcement learning, 2016. URL https://arxiv.org/abs/1606.03137.

Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell. The off-switch game, 2017. URL https://arxiv.org/abs/1611.08219. First submitted in 2016; revised version dated June 2017.

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, 2021. URL https://arxiv.org/abs/2106.09685.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017. doi: 10.1073/pnas.1611835114. URL https://arxiv.org/abs/1612.00796.

Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges, 2019. URL https://arxiv.org/abs/1907.00182.

National Institute of Standards and Technology. McNemar test. Dataplot Reference Manual, n.d. URL https://www.itl.nist.gov/div898/software/dataplot/refman1/auxillar/mcnemar.htm. Accessed 11 September 2026.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024. doi: 10.1038/s41586-024-07566-y. URL https://www.nature.com/articles/s41586-024-07566-y. Author correction published in 2025: doi:10.1038/s41586-025-08905-3.

Armando Solar-Lezama, Christopher Grant Jones, and Rastislav Bodík. Sketching concurrent data structures. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2008. URL https://people.csail.mit.edu/asolar/papers/Solar-LezamaJB08.pdf.

Richard S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 216–224, 1990. doi: 10.1016/B978-1-55860-141-3.50030-4. URL https://mlanthology.org/icml/1990/sutton1990icml-integrated/.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning, 2022. URL https://arxiv.org/abs/2203.14465.

A Evidence and data coverage

The companion archive contains 23,099 regular files (12.009 GB before compression) from the listed folders. A complete file list and study catalogue accompany the paper. These totals include copies and overlapping records, not independent measurements.

A.1 Evidence index

E1. Implementation Code for state updates, prediction-error scores and text representations; design notes.

E2. Mathematics with new numerical values 4 September report; claim-transfer-clean-2026-09-03T170409Z protocol, verdict, data manifests and AI review.

E3. Harder mathematics 5 September report; decisive-formal-2026-09-05T101400Z verdict and review. The earlier inactive-adapter run remains invalid.

E4. Answer-format test 5 September report; formal-interface-2026-09-05T150825Z protocol and verdict.

E5. Continued learning 7 September report; private-learning/dreams/20260906T214327762030Z plan, test questions, 168 measurements and update decision.

E6. Learning and using world rules 5-6 September reports; decisive-worlds-2026-09-05T192730Z, decisive-ruleuse-2026-09-05T213200Z and decisive-acquisition-2026-09-05T221330Z verdicts.

E7. Error correction and decisive-learning history Error-correction counts with an unresolved study decision; five invalid August attempts; v9-v10 preparation records; v11-v15 saved pilot results and stop records.

E8. Earlier programme records Complete available research album, data roots and laboratory records; LAWS.md and ROADMAP.md for earlier observations and stage measurements.

E9. Observational exchange plots Frozen numeric export from LoopseedSite/crates/record/data/skin.json, exported 5 September 2026. exchange-audit.json identifies the file contents, summary calculations, counts and limits. exchange-export-source.py preserves the exporter; build_exchange_figures.py checks the export and regenerates the figures without a database or model.

A.2 Archive folders

Source group	Files	MB
`Loopseed-decisive-v10-exec/data`	114	67.7
`Loopseed-decisive-v11-exec/data`	168	83.1
`Loopseed-decisive-v12-exec/data`	196	96.8
`Loopseed-decisive-v13-exec/data`	270	117.4
`Loopseed-decisive-v14-exec/data`	189	88.6
`Loopseed-decisive-v15-exec/data`	159	81.4
`Loopseed-decisive-v9-exec/data`	92	104.3

Source group	Files	MB
Loopseed/data/dyad	15	0.0
Loopseed/data/fish	33	10.5
Loopseed/data/fish-b	1	0.0
Loopseed/data/fish-c	1	0.0
Loopseed/data/fish/classrooms	10	4.1
Loopseed/data/fish/experiments	2,775	281.6
Loopseed/data/fish/labs	18,327	10950.9
Loopseed/data/fish/manual-close	48	0.0
Loopseed/data/fish/night	39	4.3
Loopseed/data/fish/speaking	8	42.7
Loopseed/data/root	0	0.0
Loopseed/data/runs	18	13.9
Loopseed/fish/album	271	48.1
Loopseed/fish/lab	365	13.8

File links are recorded without copying their targets. Authentication files, operating-system metadata and generated caches are excluded. The archive preserves files available in the listed working copies. It cannot recover missing earlier copies, and its databases were not all captured at one instant.

Paper and figure sources. The source bundle includes `manuscript.tex`, `references.bib`, vector figures, aggregate `research-record.json`, numeric `exchange-scores.json` and verification scripts. The raw-data archive is separate. The figure builders check their inputs and recreate the plots without training or generating model replies.