AUTARK Academic V3

Test Methodology and Anti-Goodhart Principles

Evidence status: 30 August 2026

1. Purpose of the test methodology

AUTARK Academic V3 is not judged merely by whether individual answers look impressive. The central question is whether we can establish, with a defensible level of confidence,

- which tasks the system actually solves;
- how reliably it completes them technically to a usable answer;
- how often the substantive answer is correct;
- which parts of the architecture contributed to the outcome; and
- whether improvements continue to hold on **new tasks that were not previously used for optimization**.

The central evaluation problem for a self-developed AI system is **Goodharting**:

> *Once a metric becomes a direct optimization target, there is a growing risk that the system will improve on that metric without the underlying general capability improving to the same extent.*

With LLM systems this can happen very subtly. Even small choices can make a benchmark look better after the fact:

- repeatedly testing the same tasks;
- changing prompts after seeing the errors;
- dropping bad cases from the statistics;
- treating technical failures as “almost correct”;
- reinterpreting an answer key after an unexpected result;
- reporting only successful model routes;
- presenting a selected failure-recovery study as if it were a new general benchmark;
- pooling different development stages into one statistic; or
- modifying the test design after seeing results until the desired outcome appears.

The AUTARK V3 methodology therefore does not try to avoid Goodharting merely by good intentions. It builds **procedural barriers** against it.

The unit of counting is itself part of the methodology. “48 tasks”, “144 task-model slots”, “288 author runs”, and “several hundred grader evaluations” refer to different things and must not be collapsed into one deceptively precise benchmark number.

2. Core principle: development and evaluation are separated

A central distinction is between two kinds of work.

Development work

Here, errors may be investigated, causes diagnosed, prompts improved, routing rules changed, or technical limits adjusted.

Examples include:

- analyzing a technical R671 failure;
- increasing a token limit;
- repairing a guard race;
- rebinding a tool path; or
- designing a new routing hypothesis.

Such work produces valuable **development evidence**, but not automatically a new general performance claim.

Evaluation

Here, the conditions of the test are fixed before execution and are not changed because of the observed outcome.

This includes:

- fresh tasks;
- frozen test sets;
- frozen ordering;
- frozen answer keys;
- frozen grading rules;
- fixed models, seeds, and prompt semantics;
- pre-defined criteria for technical failures;
- pre-defined adjudication rules; and
- a clear closure state.

This separation is one of the most important Anti-Goodhart barriers.

3. Freeze before exposure

Before a genuine evaluation run begins, the central components of the test are **frozen**.

A freeze typically covers:

- task set;
- ordering;
- answer keys;
- rubrics;
- model assignments or routing logic;
- graders;
- seeds;
- retry rules;
- technical-failure criteria;
- statistical analysis; and
- where applicable, selection or deployment thresholds.

The idea is simple:

> *Anything changed after the answers have been seen must no longer be presented as if it had been fixed before measurement.*

A freeze is therefore stored as its own artifact and cryptographically bound with SHA-256.

4. Fresh holdouts instead of repeated optimization on the same tasks

A result becomes less informative the more often the same tasks have been examined during development.

For consequential evaluations, AUTARK therefore uses **fresh holdouts**:

- new tasks;
- new variants;
- family-wise separated tasks; or
- prospectively generated test sets.

An old test can still be useful for regression testing or diagnosis. It is simply no longer an independent performance demonstration.

For that reason, the final 48-task test was not another pass over the old benchmark. It was a new frozen holdout. It was the final independent Production check, not the first or only test of the system. A substantial development and diagnostic chain - First Pass and Extensions B/C/A - preceded it.

5. Scale of the empirical evidence: 48 is not the total number of tests

The 48 new tasks highlighted in public summaries belong to the final fresh end-to-end holdout. They are scientifically important precisely because the earlier tasks had already been examined extensively and could no longer provide an independent confirmation of the same architecture.

Before this final holdout, however, a substantially larger experimental chain had already been run. Its scale must be reported using several distinct counting units.

5.1 Author-side evaluation slots before the final holdout

Experiment	Design	New author-side evaluation slots / runs
First Pass	48 academic tasks x 3 local models	144
Extension B	the same 48 tasks through the real Operational System S	48
Extension C	24 FULL runs; on a diagnostic 8-task subset an additional 8 NO_RETRIEVAL + 8 NO_FORMAL_TOOLS	40
Extension A	28 selected task-model units; one R1 and one R2 self-revision each	56
Total before the final fresh holdout		**288**

The number **288** is therefore a useful conservative count of the planned or scientifically evaluated author-side task runs/slots in First Pass + B + C + A. It is **not** a benchmark N of 288 independent tasks. Many runs use the same underlying tasks under different models, routes, or interventions.

The correct statement is not:

> *"AUTARK was tested on 288 independent tasks before the final test."*

but rather:

> *"Before the final fresh 48-task holdout, there had already been 288 controlled author-side evaluation slots or runs across baseline, routing, tool/retrieval, and self-correction experiments."*

5.2 The First Pass alone was much larger than "48 tests"

The First Pass consisted of **48 tasks x three models = 144 task-model slots**. The three frozen result vectors were:

- R1-671B: 45 C / 1 I / 1 U / 1 T;
- Qwen3-235B: 42 C / 2 I / 4 U / 0 T;
- DSV4: 40 C / 5 I / 1 U / 2 T.

Those 144 slots already produced 141 gradeable answers. Every such answer was passed to both primary graders J1 and J2; 89 cases were additionally flagged for GLM adjudication under the frozen disagreement rules, and 85 of them received a valid GLM judgment. Thus, even in the First Pass alone, grading work already amounted to several hundred evaluation units in addition to the solver slots.

5.3 Physical LLM requests are more numerous again

A scientific task slot is not necessarily identical to one HTTP or LLM request. Frozen technical retry rules may permit several physical requests for one scientific slot without turning those requests into multiple academic datapoints.

A concrete example: 48 DSV4 task slots were evaluated in the First Pass, but the documented attempt counts were:

- 40 slots with 1 attempt;
- 5 slots with 2 attempts;
- 1 slot with 3 attempts;
- 2 slots with 4 attempts.

This amounts to **61 physical DSV4 author requests for 48 scientific DSV4 slots**. For this reason alone, a single global number labelled “LLM calls” would be misleading unless the counting rule were explicitly defined.

5.4 Evaluation continued after the final holdout

The final 48-task holdout was also not the end of the empirical work. It was followed, among other things, by:

- RM16: 24 fresh mathematics/physics tasks x 2 matched author lanes (R70 and R671) = 48 additional author-side evaluation runs;
- RM17.1.1A: so far, 2 selected failure-recovery runs under a higher completion budget.

For the stages explicitly enumerated here, the conservative count through RM17.1.1A is therefore:

288 + 48 final holdout + 48 RM16 + 2 RM17 = 386 author-side evaluation slots/runs.

These 386 are still not 386 independent tasks, nor are they the total number of physical model inferences in the project. They exclude, for example:

- primary-grader and GLM inferences in the extensions and later tests;
- technical retries;
- RM15 calibration and revalidation runs;
- safety, lifecycle, and fault-injection canaries;
- planner, tool, and Lean re-attestations; and
- technical preflights with no academic task exposure.

The actual number of local model inferences is therefore substantially higher. An exact global call count would only be meaningful after a separate artifact-wide audit.

5.5 Why this distinction is itself Anti-Goodhart

It would be tempting to cite the largest available number and turn it into an impressive benchmark size. That would be methodologically wrong.

AUTARK therefore distinguishes at least:

1. unique tasks;
2. task-model or task-system evaluation slots;
3. author/solver generations;
4. physical requests including technical retries;
5. grader/adjudication inferences; and
6. fresh independent holdout tasks.

The final value **44/48 = 91.67%** refers only to the fresh final 48-task holdout. The earlier 288 author-side runs form development and mechanism evidence; they are not pooled with the holdout to manufacture a retrospective overall score.

This separation blocks another form of Goodharting: test volume can be reported transparently without presenting reused tasks or diagnostic interventions as additional independent benchmark observations.

6. Family-wise design against subtle leakage effects

A problem with LLM benchmarks is that two tasks may look superficially different while sharing the same mathematical, physical, or argumentative structure.

AUTARK therefore uses **task families**.

In the final 48-task test:

- 24 families;
- two variants per family;
- 48 tasks in total.

Statistical uncertainty is not computed as though all 48 answers were mutually independent.

Instead, a **family-clustered bootstrap** was used.

This is more conservative than a naive bootstrap over individual tasks because two variants from the same family can share failure causes.

7. Blind grading

Scientific grading should know as little as possible about which system or route produced an answer.

Answers are therefore transferred into a **blind corpus**.

The graders receive:

- the task;
- the answer; and
- the grading rubric,

but not the identity of the author model, unless that information is genuinely required for grading.

This reduces the risk that a grader unconsciously reasons:

> *"This came from the large 671B model, so it is probably right."*

or:

> *"This came only from the 35B model, so I should be more skeptical."*

Blindness is not perfect protection, but it is an important barrier against expectation effects.

8. Multi-stage grading instead of a single judgment

AUTARK does not use one LLM grader as a "truth machine".

The typical structure is:

- J1: primary grader;
- J2: second grading path;
- GLM: selective adjudication for disagreement or technical failure.

The rules are fixed in advance.

Example from RM16:

- both primary graders valid and in agreement -> shared decision;
- both valid but disagree -> GLM adjudication;
- one primary grader technical, one valid -> GLM and the valid grader must agree;
- both primary graders technical -> unresolved;
- GLM alone is insufficient when the frozen rule requires two valid paths.

This prevents the third grader from being invoked ad hoc merely when the result is inconvenient.

9. Separating substantive errors from technical failures

AUTARK distinguishes four outcome classes:

- **C - Correct**;
- **I - Incorrect**;
- **U - Unresolved**;
- **T - Technical**.

This distinction is central.

A technical failure is not the same thing as an incorrect answer.

Examples of `T` include:

- no final answer;
- a request failing technically;
- the model hitting a defined output limit; or
- a guard/runtime failure preventing a usable answer.

`U` means:

> *An answer exists, but the frozen grading rules do not permit a sufficiently reliable objective decision.*

Neither class is imputed after the fact.

This prevents two common forms of benchmark cosmetics:

1. simply dropping technical failures from the denominator; and
2. retrospectively counting ambiguous cases as “probably correct”.

10. End-to-end metrics versus conditional quality

Large reasoning models can produce exceptionally strong answers - **if they finish**.

We therefore separate two questions.

Conditional quality

How good is the model among cases for which a usable answer exists?

End-to-end performance

How often does the system produce a correct, usable answer under the actual test conditions?

RM16 demonstrates why this distinction matters.

R671:

- 13/15 of the usable answers correct;
- but only 15/24 technically usable.

R70:

- lower conditional correctness;
- but 20/24 technically usable.

For an automatic router, end-to-end performance matters because it does not know at launch time whether the larger model will finish this particular task.

11. Prospective selection rules

A routing decision is not invented after the test.

Before unblinding RM16, the conditions under which an automatic R671 selector could be authorized were fixed in advance.

The frozen gates included, among other things:

1. enough additional R671-only correct cases;
2. more R671-only C than R70-only C;
3. no more than one additional technical failure; and
4. no systematic resource disruption.

After unblinding:

- R671-only C = 4;
- R70-only C = 3;
- R671 T = 9;
- R70 T = 4.

The reliability gate therefore failed clearly.

The automatic R671 selector was **not authorized**.

The important point is:

> *The deployment rule was not relaxed after the result became known.*

12. Immutable closure

Once a benchmark or experimental stage is closed, it remains **closed**.

That means:

- no retrospective score changes;
- no regrading of valid answers;
- no replacement of the answer key;
- no rerun of the same author task to obtain a better outcome; and
- no retrospective reclassification of an inconvenient failure.

A closed result remains historically true even if later work provides more information.

13. Example: the convention-sensitive phi^3 case

In the final 48-task test, four answers were officially classified as `I`.

For one of these cases - phi^3 theory in six dimensions - a later post-closure audit showed that the frozen answer key was convention-sensitive and probably too narrow.

Nevertheless, the official benchmark remained:

- 44 C;
- 4 I;
- 0 U;
- 0 T;

not 45/48.

This is a particularly clear Anti-Goodhart example:

> *The benchmark is not retrospectively improved merely because a later analysis provides a good reason to interpret an old case more favorably.*

The later analysis is documented separately.

14. Append-only errata rather than rewriting history

Technical errors in test harnesses are not deleted from the history either.

If a runner contains a defect, the record contains:

- a historical attempt;
- a documented `TECHNICAL_STOP`; and
- a new successor or erratum run.

Example:

RM17.1 Attempt 1 stopped in preflight because of a dictionary-key mismatch in the runtime-binding check.

The result was not deleted or relabeled.

Instead, RM17.1.1 was built to correct only the identified harness cause.

This preserves a trace of:

- what originally happened;
- why it happened; and
- exactly what was changed in the successor.

15. Minimal-change successors

An erratum successor should repair as little as possible beyond the demonstrated defect.

Example RM17.1.1.

Permitted changes:

1. key normalization in the runtime-SHA check;
2. direct use of the canonical RM16.3.6 REQUEST files instead of heuristic search.

Not changed:

- scientific intervention;
- task set;
- models;
- retry semantics;
- resource policy;
- grading;
- Production architecture.

In addition, unchanged runtime and resource functions were compared with the parent at the AST level.

This reduces the chance that what appears to be a technical fix quietly changes the scientific experiment at the same time.

16. Exactly-once semantics

For prospective author captures, the protocol often requires:

- exactly one attempt per task;
- no substantive retries; and
- no silent reruns of poor answers.

A retry can itself become an optimization mechanism:

> *“If the answer is bad, ask again.”*

But a system that needs three attempts is not the same system as one that solves the task on the first attempt.

Retries are therefore allowed only where they were explicitly part of the frozen protocol.

17. No salvage of internal reasoning tokens as the final answer

Reasoning models can produce very long internal reasoning traces without ever reaching a final answer.

Such a reasoning trace is **not retrospectively salvaged as the answer**.

Example RM17:

A selected R671 case produced, at `max_tokens=24576`:

- a very long reasoning trace;
- `finish_reason=length`;
- but no final answer.

The case remains technically unrecovered.

This blocks another form of retrospective outcome improvement.

18. Failure-set diagnostics are not sold as new benchmarks

After RM16, nine technical R671 failures were selected for targeted investigation.

This is deliberately a **selected failure set**.

The question is:

> *How many of these known failures recover if only the completion-budget parameter changes?*

That is a mechanism diagnostic.

It is **not** an estimate of general R671 reliability.

The statement

> *“1/2 recovered, therefore reliability is 50%”*

would be methodologically wrong.

The correct statement is:

> *Of the first two deliberately selected earlier failure cases, one technically recovered under the higher budget and one did not.*

This separation between

- a general benchmark;
- a prospective matched comparison; and
- a selected failure diagnostic

is a central Anti-Goodhart component.

19. Different studies are not pooled

RM15, RM16, and RM17 answer different questions.

RM15

Question:

How well does the frozen Production architecture work end to end on a fresh 48-task holdout?

RM16

Question:

Would an automatic R671 path deliver enough additional performance relative to R70 on a prospective matched set without losing too much reliability?

RM17

Question:

How much of the observed R671 technical-failure set can be explained by a larger completion budget?

These results are not combined into one success rate.

20. Matched designs for model comparisons

When two models are compared, they should receive the same tasks whenever possible.

RM16 used:

- 24 fresh tasks;
- 12 families x 2 variants;
- R70 and R671 on the same tasks.

This permits **paired analysis**.

More informative than raw total scores are categories such as:

- both correct;
- R671 only correct;
- R70 only correct;
- neither correct.

In closed RM16:

- both C = 9;
- R671-only C = 4;
- R70-only C = 3;
- neither C = 8.

The net difference was therefore small.

The exact paired test gave:

- directional p = 0.5;
- two-sided p = 1.0.

Thus there was no statistically compelling evidence of an end-to-end advantage for R671 in that experiment.

21. Cluster bootstrap rather than false precision

In addition to exact paired tests, AUTARK uses family-cluster bootstraps.

Why?

If tasks from the same family share similar failure causes, they should not be treated as fully independent observations.

In the final 48-task test, the family-clustered bootstrap gave a 95% interval for accuracy of:

- **81.25% to 100%**.

The point estimate of 91.67% is therefore not presented as a universal, highly precise property of the system.

22. Auditability through cryptographic binding

Important artifacts are bound with SHA-256.

Depending on the stage, these include:

- task sets;
- contracts;
- runners;
- blind corpora;
- grading results;
- closure artifacts;
- result ZIPs; and
- source bindings.

Many ZIP archives additionally contain an internal `SHA256SUMS.txt`.

This makes it possible to check later:

- whether the exact same artifact is being examined;
- whether files have changed; and
- whether a claimed result is actually linked to the claimed freeze.

Cryptographic binding does not replace scientific methodology, but it substantially improves the traceability of the process history.

23. Source binding and runtime identity

Freezing the task set is not sufficient for a system test.

The software that actually runs must also be identified.

AUTARK therefore checks SHA-256 values for, among other things:

- the Production router;
- the start wrapper;
- the thermal guard;
- llama-server.

If an expected source hash does not match, the run stops.

This prevents a result from being generated silently with a different runtime than the one specified by the test design.

24. Resource conditions are part of reproducibility

For very large local models, resource conditions are not mere operational details.

They affect:

- technical reliability;
- latency;
- possible concurrency;
- model residency; and
- potentially the behavior of the overall system.

Heavy-model protocols therefore freeze conditions such as:

- swap must be 0 before startup;

- R671 starts only with very high free-memory headroom;
- other heavy models must be cold;
- thermal states are monitored;
- heavy models are serialized; and
- after completion, a cold state must be demonstrated.

A scientific run performed under uncontrolled resource conditions is not silently treated as equivalent to a controlled run.

25. No causal claim without execution evidence

AUTARK distinguishes between:

- a tool being installed;
- a tool being integrated;
- the router planning tool use;
- a gate accepting tool use;
- the tool actually being executed; and
- the tool execution causally influencing the final answer.

These stages must not be conflated.

In the final 48-task benchmark, formal tools were integrated and the tool architecture had been validated separately.

The benchmark itself, however, did **not** prove actual CAS/Lean execution for the relevant answers.

Therefore 44/48 is not used to claim:

> *"The formal tools improved the score."*

That would be a causal claim without sufficient execution evidence.

26. Negative evidence remains visible

An important Anti-Goodhart rule is:

> *A negative finding is not removed from the project history merely because a better successor later exists.*

Examples:

- failed harness attempts remain documented;
- Technical Stops remain Technical Stops;
- a rejected R671 selector remains rejected;
- RM16 is not retrospectively "repaired" by later RM17 diagnostics;
- a closed `T` remains `T` in the old experiment even if the same task succeeds later under different conditions.

This preserves what the system actually did **at a particular historical time under particular conditions**.

27. Why Anti-Goodhart is more than "not cheating"

Goodharting need not be intentional.

It often emerges precisely from conscientious development work:

1. An error is observed.
2. The error is understood better.
3. The system is improved.
4. The same task is tested again.
5. The result is better.

6. The improved case begins, often unconsciously, to be treated as evidence of a general improvement.

Every individual step is reasonable.

The methodological error appears only when **development evidence and independent evaluation are confused**.

AUTARK V3 therefore tries to keep those categories institutionally separate.

28. Practical Anti-Goodhart rules in brief

The main rules can be summarized as follows:

1. **Freeze before exposure.**
2. **Fresh holdouts for consequential performance claims.**
3. **No retrospective changes to valid answer keys or scores.**
4. **Keep C/I/U/T strictly separate.**
5. **No imputation of U or T.**
6. **Blind grading where possible.**
7. **Multiple grading paths with pre-defined adjudication.**
8. **No silent retries.**
9. **Do not interpret selected failure sets as general benchmarks.**
10. **Closed experiments remain closed.**
11. **Append-only errata instead of rewriting historical results.**
12. **Minimal-change successors.**
13. **Bind runtime and source hashes.**
14. **Make causal tool claims only with execution evidence.**
15. **Keep negative results visible.**
16. **Use fresh prospective holdouts before any new Production authorization.**

29. What this methodology does not guarantee

Even a strict protocol does not eliminate all uncertainty.

It does not prevent:

- small sample sizes;
- flawed rubrics;
- grading errors by multiple graders;
- undetected benchmark leakage;
- dependence between model families used as graders;
- limited representativeness of the tasks;
- hidden correlations among task families; or
- the fact that academic tasks cover only one part of what one might call intelligence.

Anti-Goodhart therefore does not mean:

> *"The benchmark is objective truth."*

It means:

> ***We systematically reduce the ways in which, after seeing the results, we could fool ourselves into believing that we tested a better system than we actually did.***

30. The strongest practical tests of this philosophy so far

The most informative evidence for the methodology is not the score itself, but what happens when results are inconvenient or unexpected.

Three examples:

Example A - 44/48 remains 44/48

Even after one official error was later recognized as convention-sensitive, the closed benchmark remained unchanged.

Example B - no automatic R671 selector

Although R671 was academically strong when it finished, it failed the pre-frozen reliability gate. The selector was therefore not authorized.

Example C - RM17 does not change RM16

If an old RM16 `T` later produces a good final answer under 24576 tokens, the original RM16 case remains `T`.

The new run answers a **new question under new conditions**.

31. Short form for public description

Scale of the evidence: The final score of 44/48 comes from a fresh, pre-frozen end-to-end holdout. It was not the entire test volume. First Pass and Extensions B/C/A had already produced **288 controlled author-side evaluation slots/runs** beforehand; adding the final holdout, RM16, and the first RM17 diagnostics gives at least **386 explicitly counted author-side evaluation slots/runs** at the evidence state documented here. This is not a benchmark N of independent tasks and does not include grader inferences, technical retries, or additional calibration/safety tests.

AUTARK Academic V3 uses an explicitly Anti-Goodhart-oriented test methodology. Consequential tests are frozen before execution: tasks, ordering, answer keys, grading rules, technical-failure criteria, and, where relevant, release thresholds. Important performance claims use fresh holdouts. Answers are graded blind where possible and through multiple grading paths; technical failures, substantive errors, and unresolved cases remain separate. Closed benchmarks are not rewritten after the fact - even when a later analysis makes an old case look more favorable. Diagnostic reruns of known failures are treated as new experiments and are not used to improve older scores retrospectively. Artifacts, runners, and results are cryptographically bound, and causal claims about tools are made only when actual tool execution is demonstrated.

32. One-sentence version

> ***The Anti-Goodhart idea is: define what counts as success first; then measure it; do not reshape the historical result afterward - instead, demonstrate every improvement in a new, separate experiment.***

33. Authoritative examples from the project so far

Final 48-task benchmark

Status:

`RM151228_PASS_FINAL_BENCHMARK_CLOSED_NO_FURTHER_BENCHMARK_INFERENCE_OR_REGRADE`

Result:

- C44;
- I4;
- U0;
- T0;
- N48;
- technical reliability 100%.

Closure SHA-256:

`2f50add048b0bec7d7ea560b7b5fcc6bf4f81916ba39e11282e9760612378e62`

RM16 matched R70-vs-R671

Result:

- R70: C12 / I5 / U3 / T4;
- R671: C13 / I2 / U0 / T9;
- automatic R671 selector: REJECT.

Closure SHA-256:

`0d825df9c98fee8c154570d1c7202c4e27d442bed3af6b3311ef6a3d41e42054`

RM17.1.1A selected-failure diagnostic

Two of the nine frozen old R671-T cases were examined under a larger completion budget:

- one technically recovered;
- one remained `finish_reason=length` at 24576 tokens.

These results do not change RM16.

Result SHA-256:

`4a175a8e3b5974b1c9d9247b081d2e76d20395b0048a07072d34bc2a749043af`

Conclusion

The AUTARK V3 test methodology is deliberately conservative.

It does not try to eliminate every uncertainty by compressing everything into one number.

Instead, it separates:

- development from evaluation;
- technical reliability from substantive correctness;
- general benchmarks from selected diagnostics;
- observed correlation from causal evidence; and
- historical results from later improvements.

A further principle concerns how test scale is reported: large call or run counts must not be represented as numbers of independent tasks. Conversely, the final fresh holdout must not be presented in a way that suggests the system had only seen 48 tests before it.

The central Anti-Goodhart principle is therefore:

> ***An improvement counts as general progress only after it has been confirmed in a new, pre-specified test that is as independent as practicable from the error that motivated the improvement.***