

Highlights

A machine-written mathematics release read against its claims: the Navier–Stokes certificate, the challenges under two kernels, and the statements their manuscripts claim*

Carlos Toledo

- 416 Lean challenges replayed under two kernels: 398 accepted by both, 18 refused for resources, none rejected.
- The Navier–Stokes blowup proof builds on the standard axioms; its statement is Clay’s; one clause of the paper’s theorem was not formal at the announced commit.
- 34 of 242 families state less than, or other than, their headline without saying so.
- A reading is not a verdict: the readings are AI-made, recorded with their second readings, and their recall was measured on planted defects.
- Every input is pinned by digest; every count names its denominator; the pre-registration and its amendments are public.

A machine-written mathematics release read against its claims: the Navier–Stokes certificate, the challenges under two kernels, and the statements their manuscripts claim

Carlos Toledo^{a,*}

^aIndependent researcher, the cert-machine project, Brazil

Abstract

In September 2026 OpenAI published 719 machine-written mathematics manuscripts in 372 families, with 416 Lean challenge statements as their formal spine, and a Lean proof of finite-time blowup for the forced three-dimensional Navier–Stokes equations. A formal certificate makes two claims: that a kernel accepts the proof, which a build decides, and that the statement proved is the theorem claimed, which no build decides. We decided the first and read the second from bytes pinned by digest, under a pre-registration amended 16 times, each amendment dated before the run it changed. Every challenge was replayed under Lean’s kernel and under a second kernel written independently of it: 398 were accepted by both, 18 were refused for resources, none was rejected. Every statement was read against the headline its manuscript claims: of 242 families, 127 match and 34 state less than, or other than, what they claim without saying so. The Navier–Stokes proof was built at its announced commit on the three standard axioms and accepted by both kernels; its formal statement is Clay’s, and one clause of the paper’s own theorem, the uniform energy bound, was not among its conclusions at that commit; it is a conclusion of the next upstream commit, decided here with one kernel. 14 gaps between what the release claims and what it checks are recorded, each pinned to a file and line; 6 are headline theorems that no checked statement carries. The readings were made by AI agents and no specialist has confirmed them; their recall was measured once, on 12 defects planted in short statements, all flagged. Nothing in the release was found

*Draft v0.1 of 2026-10-21, built from repository commit 16849460e608; it will be archived with the next release of the repository, under the concept doi:10.5281/zenodo.22225860 (the previous version is in release v2026.10.18, doi:10.5281/zenodo.23446020). Every measured number and count is a macro read at build time from the records named in the Data availability statement; a record that no longer supports a sentence stops the build. Companion to the method paper (Toledo, 2026, the method paper); the Navier–Stokes certificate is audited line by line in a separate note (Toledo, 2026, the Navier–Stokes note).

*Corresponding author.

Email address: carlos@carlostoledo.co (Carlos Toledo)

false. What is claimed is the two decisions and the reading, each with its scope; what is not claimed is the mathematics.

Keywords: machine-generated mathematics, formal proof, autoformalization, independent checking, Lean, Navier–Stokes, certificates, reproducibility

1. Introduction

Formal proof has become the currency of machine-made mathematics. Systems that search for proofs in Lean now reach Olympiad problems (Hubert et al., 2025; Ren et al., 2025; Wang et al., 2025; Lin et al., 2025), and a release of research-level results arrives with a Lean development as its warrant. In September 2026 OpenAI published a mathematics release of 719 manuscripts in 372 families (OpenAI, 2026a) and, beside it, a Lean 4 repository proving finite-time blowup for the forced three-dimensional Navier–Stokes equations against a statement that DeepMind’s Formal Conjectures project had written months earlier (OpenAI, 2026b; Google DeepMind, 2026). The release describes itself as “Partial progress”, with review “unchecked”, and counts 300 of its 719 manuscripts (42%) as formalized at its top line. Its formal spine is 416 Comparator challenges: Lean statements, written by the claimant, that a proof must satisfy to be accepted (Lean FRO, 2026).

A formal certificate makes two claims, and only one of them is a computation. That the kernel accepts the proof is decided by building it at the pinned commit, asking the kernel which axioms the result depends on, and replaying the proof under a second kernel written independently of the first. That the statement proved is the theorem claimed is not decided by any build: a translation that compiles may still not be the theorem intended, which is the general point an independent paper on the same release argues (Bastounis et al., 2026), and which the autoformalization literature has measured on benchmarks (Wu et al., 2022; Lu et al., 2024; Liu et al., 2025; Poiroux et al., 2025). This paper decides the first claim for every challenge of the release and for the Navier–Stokes proof, and reads the second, hypothesis by hypothesis, for every challenge against the headline its manuscript claims. The reading issues no verdict of its own.

The method is the one of the cert-machine project (Toledo, 2026, the method paper): a claim is read as a statement over pinned bytes, decided independently of the claimant with no code of the claimant’s in the trust path, and recorded with one of three verdicts and the object that justifies it, where a refusal is a result that names its reason. Here the deciders are the two kernels and Comparator, which the claimant itself used; what this project adds is the pre-registration, the second kernel switched on for every challenge where the release enabled it for 2, the reading of every statement

against its claim, the measurement of that reading’s recall on planted defects, and the record.

Contributions. (i) The kernel lane: every challenge replayed under two kernels at the release’s pins on hosted runners and then on a rented machine under a pre-registered cap, with the refusals counted by reason and never summed with rejections (there were none). (ii) The Navier–Stokes certificate: built, its axioms asked of the kernel, replayed by the second kernel and by Comparator, its statement read against Clay’s wording and the paper’s theorem, the next upstream commit decided. (iii) The statement lane: 242 families read against their headlines with a fixed vocabulary of 6 words, every family not read as matching or declared read a second time, the less severe word recorded. (iv) A measurement of the readings’ recall on 12 planted defects. (v) 14 gaps between what the release claims and what it checks, pinned to file and line with the ask that would close each. (vi) The cost and the hosted profile of every run.

Outline. Section 2 gives the materials and the pre-registration, Section 3 the four lanes, Section 4 the results, Section 5 what they show and what they do not, and Section 6 concludes. The reproduction table is in Appendix A.

2. Materials

The release. The release at commit fd4aeeb2 holds 719 manuscripts, 372 families and 416 challenges, with 3 manuscripts withdrawn by the claimant; manuscripts, families and challenges are three denominators and are never mixed in one count. Each challenge is a Lean file with a statement, a configuration naming the definitions the proof may leave as holes, and, in 2 of the 416 configurations, the second kernel switched on (412 off, 2 absent). A manuscript’s headline is the sentence the release’s contents file writes for its family; a scope note beside each formalized family says what the formalization leaves out.

The Navier–Stokes repository. 24 files were pinned on 2026-09-09, the day after the announcement: the repository at commit 8937a8f4 (2,486 files, 618,762 lines, toolchain v4.34.0-rc2, Mathlib at 85e3a25e), the 166-page manuscript and the 57-page companion on Euler, the announcement, and the Formal Conjectures statement the proof targets, added on 2026-05-15 (#1457) and last changed on 2026-07-27, the revision the claimant pinned. Every file is named by path and SHA-256 and re-hashed when it is used.

The machines. The Navier–Stokes build ran on a laptop with 8 cores and 16 GB of memory. The challenge replays ran on GitHub’s hosted runners (4 vCPU, 16 GB, 6 h per job), one job per challenge, and then on one

rented CPU pod (32 vCPU, 256 GB, at \$1.76 per hour, approved on 2026-10-10) and a second (32 vCPU, 64 GB, \$0.96 per hour) within a \$10 cap approved on 2026-10-12. The pods held no credential: each cloned the public repositories, built the tools from source at their pins and served its facts read-only.

The pre-registration. The audit was registered on 2026-10-10, before any challenge was run: each lane’s membership, its decider, the words it may return and the second-reading rule (Toledo, 2026, the openai/math report). It was amended 16 times, every amendment dated and placed before the run it changed.

3. Method

3.1. Two claims of a formal certificate

A formal certificate is read as two claims over a box, in the grammar of the method paper. The kernel’s claim—that the proof is accepted—is decided: the development is built at the pinned commit with the toolchain and library the claimant pinned; the kernel is asked which axioms each main declaration depends on; the proof is exported and replayed under a second kernel; and Comparator (Lean FRO, 2026), the Lean FRO’s judging tool, compares the theorem proved with the challenge statement, name by name, universe by universe, type by type. The verdict is `CERTIFIED` when the theorem states exactly the challenge, depends only on `propext`, `Classical.choice`, `Quot.sound`, and both kernels accept, and `REFUSED` when a resource ran out before a kernel answered; a kernel that rejected would be `REFUTED`, and none did. The statement’s claim—that the theorem proved is the one claimed—is read: the formal statement is set beside the headline, the abstracts the scope note names and the scope note itself, and every difference is recorded. A reading is not a certificate and is never called one. Throughout, manuscripts, families and challenges are never mixed in one count, a lane’s count is stated over the rows it decided, refusal kinds are never summed into one total, and every number is a macro written from a record by a generator that refuses when the record no longer supports the sentence quoting it.

3.2. The kernel lane

The decider is Comparator at commit d03acab1 (tag v4.34.0, built with the release’s toolchain v4.34.1), the exporter `lean4export` at 076e8e57, the sandbox `landrun` at 811cfff5, and the second kernel `nanoda` at 3a240721 (ammkrn, 2026), a type checker for Lean 4 export files written independently of Lean’s kernel, in the line of independent checkers for the language (Carneiro, 2024). Three changes to the release’s own configuration were

declared before the run. The second kernel was switched on for every challenge. The 10 configurations that declare definition holes were run a second time with the holes emptied, because Comparator compares a hole’s name, universe levels, type and safety and never its body, so the second run decides whether the theorem is proved about the definitions the challenge file displays. And the second kernel’s worker-thread stack, a hard-coded 16 MiB, was raised to 1 GiB, the only byte of its source changed; a later amendment added a patch of two reduction rules, so that four builds of the second kernel appear in the record (Section 4). The challenge bytes and the configuration are checked against the census by SHA-256 before every job.

3.3. *The statement lane*

Every family with a challenge is read against three sources: the headline the release writes for it, the abstracts of the manuscripts its scope note names, and the scope note. The reading returns one of 6 words. MATCHES: the Lean statement states the headline, up to the formalization’s unavoidable encoding choices, each named. NARROWER—DECLARED: it proves less, and the scope note says what is left out. NARROWER—UNDECLARED: it proves less, and neither the scope note nor the challenge file says so. NARROWER—UNLINKED: it proves less than the family’s headline, and every missing clause comes from a manuscript the scope note does not link, so the formalization never claimed it. DIFFERENT: the statement neither implies nor is implied by the headline. SUPPORT-ONLY: the claimant declares the challenge a supporting result. The unit of counting is the family: a family’s challenges are read together, and the family carries one word. Every family a first reader did not read as MATCHES or NARROWER—DECLARED was read again by a second reader who saw the first reading; where the two disagreed, the less severe word was recorded and both are shown. The readers are AI agents working from the written brief of the pre-registration; no specialist in the fields concerned has confirmed any reading, and the paper reports the readings as readings.

3.4. *The finite-core lane*

A manuscript whose headline, or a stated component of it, is witnessed by a finite explicit object—a graph, a table, a configuration, a polynomial identity—is a claim about that object, decided exactly wherever the object is published. A census of all 719 abstracts by 3 readers on disjoint slices fixed 66 rows over 65 manuscripts in three tiers (47 A, 4 B, 15 C) and was appended to the pre-registration before any row was decided; 92 near misses are recorded with the reason. Each row is decided by a clean-room re-decision sharing no code with the release, in exact rationals, integers, finite-field arithmetic or interval enclosures as the object requires, with red controls that perturb the witness and must refute; a row is CERTIFIED only

when the whole finite certificate its manuscript rests on is decided, and one whose witness is asserted but not published is NEEDS DATA, the file named.

3.5. *The recall of a reading*

A reading issues no verdict, but its misses can be counted. From the 30 families read as MATCHES whose one challenge is at most seventy lines with no definition hole, 24 were drawn with a seed (20261019), and in 12 of them one line of the statement was changed so that it states less than the headline, or something else, in one of 5 kinds: a constant or exponent weakened, a quantifier’s domain restricted, a hypothesis added, a conclusion weakened, a direction reversed; the scope note was left silent. The other 12 packets are the release’s bytes. 4 fresh AI readers, not told that anything was planted, read every packet blind under the pre-registered brief in 2 pipelines, with the protocol’s second reading; every tool call they made (42 in all) is kept and audited, and none opened a file outside its packets. The measurement’s 6 limits are stated with it (Section 5).

4. Results

4.1. *The Navier–Stokes certificate*

The repository was built at commit 8937a8f4 over 2,486 modules of its own (11,251 with the library it imports, the other 8,765 taken from the pinned build cache) in 140 minutes of build time over two legs of 8 and 4 threads (3 h 17 min of wall-clock), and the kernel reported `propext`, `Classical.choice`, `Quot.sound` and nothing else for every main declaration asked. Comparator, with the exporter and the second kernel, accepted both challenge theorems, the periodic and the whole-space statement, in 13 and 17 minutes; a non-vacuity file of 7 declarations compiled against the result with no error. The reading set the formal statement beside Clay’s wording (Fefferman, 2000) and Mathlib’s definitions in 9 rows, 7 the same and 2 equivalent, and found it to be Clay’s alternatives (C) and (D) with nothing added to the competitor class and nothing dropped from the data conditions. It set the paper’s Theorem 1.1 beside the formal conclusions in 7 rows and found 2 clauses of the paper’s theorem absent from them at that commit, among them the uniform energy bound $\sup_{0 \leq t < 1} \|u(t)\|_{L^2} < \infty$, whose lemma was proved and used by nothing. It re-derived 6 steps of the manuscript, recorded 5 slips, none affecting a conclusion, and named 13 weak points. The record’s word at 8937a8f4 is PARTIAL: the kernels’ claim CERTIFIED by two kernels, the statement Clay’s on the reading, the energy clause named in the scope sentence.

The next upstream commit, f9e8bc5b of 2026-09-10, the day after the audit pinned the repository (188 files changed, 25,143 lines added), was decided on 2026-10-06 with one kernel: an incremental rebuild of the 486 changed or dependent modules under the pinned toolchain in 25 minutes, the

other 2,173 kept from the pin’s build, the same axioms for all 7 declarations asked. The clause is now a conclusion of `theorem_1_1`, whose statement is the claimant’s own, read here (11 rows, 9 the same, 2 equivalent) and not judged by Comparator; the exporter, the second kernel and Lake abort at start on this machine’s operating system at that toolchain, so neither was re-run, and the challenge files are byte-identical to the pin’s. The record’s word is CERTIFIED with that scope, recorded 2026-10-05, the PARTIAL of 2026-09-09 kept as its history. The note (Toledo, 2026, the Navier–Stokes note) gives the clause-by-clause tables, the five section memos, the obstructions the manuscript meets, the probes re-run, and the 12 dates of the claim as the three parties state them, adjudicated nowhere.

4.2. The challenges under two kernels

All 416 challenges were decided: 398 CERTIFIED by both kernels, 18 REFUSED, and none rejected by either kernel (Table 1). The second kernel accepted 267 as built, 127 with its stack raised to 1 GiB, and 4 with the patch that does two reduction rules as Lean’s reference kernel does. The 18 refusals have three reasons, counted apart: 2 stopped in the export-or-replay stage with the second kernel’s stack overflowing at every build; 13 jobs were terminated by the hosted runner before a result was uploaded, for memory or time; and 3 (ErdosReciprocal, MUBSix, PlanarPacking) stopped before Comparator ran. The refused statements run from 14 to 6,649 lines; the longest certified one has 19,170. Under the pre-registration’s amendment of 2026-10-12, the three that never reached Comparator were sent one at a time to the second rented pod within the \$10 cap, and the 15 the runners terminated for memory or the second kernel refused were not re-run, by the operator’s choice of the day before; what did not finish at the cap stays REFUSED. The record carries the hosted runs; the three remain REFUSED in it. Of the 10 configurations with definition holes, the second run with the holes emptied found 7 with the same bodies and 3 whose bodies differ from the displayed definitions; the transfer check that follows found 6 theorems that transfer to the displayed definitions, 2 that do not definitionally, and 2 not decided. 63 displayed definitions are never compared by Comparator, which is one of the gaps of Table 3.

4.3. The statements against their headlines

Of 242 families with a challenge (416 challenges, 92 families with more than one), 127 MATCHES; 34 state less than, or other than, what they claim without saying so (11 DIFFERENT, 23 NARROWER—UNDECLARED); 38 are NARROWER—DECLARED; 39 are NARROWER—UNLINKED; and 4 are SUPPORT-ONLY (Table 2). The second readings covered 77 families and changed the word of 40; all but one of the changes moved a family into the category the first pass did not use, narrower and not linked to a claim, which the pre-registration added by amendment, and one was a disagreement on

	count	of
challenges decided	416	416 in the census
CERTIFIED by both kernels	398	416
REFUSED for resources	18	416
second kernel’s stack overflowed, every build	2	18
job terminated by the runner (memory or time)	13	18
job stopped before Comparator ran	3	18
rejected by a kernel	0	416
second kernel enabled by the release	2	416
second kernel accepted: as built / stack 1 GiB / patched	267 / 127 / 4	398
runs recorded (both kernels in the run)	500	(414)
wall seconds per timed run, min / median / max	18 / 587 / 12,453	437 runs
peak memory per timed run, min / median / max (GiB)	1.6 / 6.8 / 13.4	437 runs
library cache fetched per run, median / max (s)	194 / 240	

Table 1: The kernel lane (`certs/openai-math-kernel.json`). A refusal names its reason and the reasons are never summed with rejections, of which there were none. The hosted profile is GitHub’s Linux runners (4 vCPU, 16 GB, 6 h), one job per challenge, as carried into the self-audit’s concordance record.

the reading itself. The less severe word was recorded, so that 62 families read as NARROWER—UNDECLARED on the first pass became 23. 51 flag readings were made on suspicious statements and 76 flags cleared. 9 families carry a word conditional on the kernel lane, their challenges among the refused.

word	families	meaning
MATCHES	127	the statement states the headline, up to named encoding choices
NARROWER—DECLARED	38	proves less; the scope note says what is left out
NARROWER—UNDECLARED	23	proves less; neither the scope note nor the file says so
NARROWER—UNLINKED	39	proves less than a headline whose missing clauses the formalization never claimed
DIFFERENT	11	neither implies nor is implied by the headline
SUPPORT-ONLY	4	declared a supporting result by the claimant
all	242	the families with a challenge; the unit is the family

Table 2: The statement lane (`certs/openai-math-statements.json`): every family’s challenges read together against its headline, the abstracts its scope note names and the scope note, by AI agents under the pre-registered brief, with the second-reading rule. A word here is a reading, not a verdict.

4.4. The finite objects

Of the 66 rows whose headline rests on a finite object, 44 were decided CERTIFIED by an exact re-decision made here, 7 were REFUSED with the reason, and 15 assert a witness the release does not publish and are NEEDS DATA, each naming the file that would decide it. Each CERTIFIED is the certificate's: the object has the property the manuscript states, and the chain from that property to the headline is the manuscript's, read and not re-derived.

4.5. The gaps

Table 3 ranks the 14 gaps between what the release claims and what it checks, found on 2026-10-10 at commit fd4aeeb2 and pinned to file and line before any fix, by the severity the audit assigned: 6 headline theorems that no checked statement carries; 3 checks weaker than shown; 4 claims that cannot be re-checked as shipped; 1 of documentation. When upstream was last checked, on 2026-10-10, its head was still fd4aeeb2, so none had been closed. Each carries the ask that would close it. Nothing in the release was found false.

4.6. The recall of the readings

In both pipelines all 12 plants were flagged, each at the planted line, 11 with the key's word; the twelfth was recorded as declared, the case the key marks arguable; and no control was flagged. Twelve of twelve bounds the rate of flagging below by 0.735 (two-sided 95% Clopper–Pearson). Counted at the audit's own threshold, where a declared narrowing is not a gap, 11 of 12 were caught, with the lower bound 0.615; in one pipeline the first reader had the key's word for the twelfth and the rule that records the less severe word downgraded it. Two plants left a clue the edit did not touch, a name and a docstring.

4.7. The cost

The Navier–Stokes build took 140 minutes on the laptop and the upstream rebuild 25. The hosted runs cost nothing beyond GitHub's allowance and are profiled in Table 1: the longest timed run took 208 minutes and the largest 13.4 GiB. The rented pods ran at \$1.76 and \$0.96 per hour, the second within a \$10 cap. The readings are not timed and are not a method.

severity	the gap
1	Hilbert’s sixteenth problem: the uniform bound is in no Lean statement
1	Kaplansky: the headline’s group is torsion-free; the Lean group is required to have torsion
1	Penrose inequalities: no Penrose inequality is stated
1	Coarse assembly: the headline is about the maximal map; the Lean is about the reduced one
1	Projected Lax: the formalized cone has an exact lift, by the family’s own third manuscript
1	Across the release: 34 of 242 formalized families state less than, or other than, what they claim
2	63 displayed definitions Comparator never compares — one of them carries ‘the plane is not 5-colorable’
2	Scope notes credit the formalization with theorems no challenge states
2	The independent kernel was switched on for 2 of 416 challenges — and cannot finish the largest as shipped
3	15 headline witnesses are asserted but not published
3	Papers cite checkers and data the release does not ship
3	Shipped checkers that stop short of what their paper claims — each step re-decided here
3	Printed numbers the text alone does not determine
4	Scope notes that are stale or contradict their own challenges

Table 3: What the release claims beyond what it checks (`corpus/openai-math/findings.json`), from 1 (a headline theorem that no checked statement carries) to 4 (documentation). The readings behind the severity-1 rows were made by AI agents (Section 5).

5. Discussion

What the exercise shows. No proof of the release was rejected and nothing was found false; the gaps are between what was claimed and what was checked. A clause present in the paper’s theorem and absent from the formal statement at the announced commit is the finite instance of the general point that a Lean verification of an autoformalized statement does not by itself establish the natural-language theorem (Bastounis et al., 2026), and of the measured fact that statement autoformalization at research level is far from reliable (Poiroux et al., 2025). 34 of 242 formal statements narrower than, or other than, the theorems beside them, with nothing in the release saying so, is the same instance at scale. The second kernel, which the release enabled for 2 challenges, accepted every proof it could finish; what it could not finish, it refused with the reason, and the refusals are reported as what they are. The commit of 2026-09-10 proves the clause; it was made the day after the audit pinned the repository, and this paper draws no causal link between the two.

Relation to the literature. Independent checkers for Lean’s export format are the line this lane stands in: a verified type checker for Lean in Lean (Carneiro, 2024), a type checker in Rust (ammkrn, 2026), and Comparator’s statement comparison (Lean FRO, 2026); the reading complements them, as a check on what the statement says rather than on whether the proof is valid. Faithfulness of autoformalization has its own metrics and benchmarks—alignment scores (Lu et al., 2024), equivalence checking against a reference (Liu et al., 2025; Poiroux et al., 2025)—which compare a machine’s translation with a human’s; here there is no reference translation, the statement is compared with the prose claim, and the comparison is a recorded reading with a measured recall rather than a score. The sociology of a world where proof checking is cheap and adjudication of what was proved is scarce has been named (Kallel et al., 2026; Avigad, 2024); this paper is one adjudication, with its scope. On the mathematics itself nothing is claimed: finite-time blowup for the forced Navier–Stokes system stands beside the averaged-equation blowup (Tao, 2016), the $C^{1,\alpha}$ Euler singularities (Elgindi, 2021) and the computer-assisted blowup of Chen and Hou (Chen and Hou, 2022, 2023), and this paper says only what the kernels accepted and what the readings found.

Threats to validity.

- *The readings.* The statement readings and the Navier–Stokes reading were made by AI agents and checked by second readings made the same way, a check on consistency and not an independent review; no specialist has confirmed them. The clause the upstream commit proves is consistent with the reading and does not confirm it.
- *The recall.* It was measured once, on 12 one-line defects planted in the audit’s easiest class—one challenge of at most seventy lines, no definition hole, read as matching—under a brief that asked for every constant, bound and domain to be checked; the audit’s long statements, eight of them over a thousand lines, and its multi-challenge families are not measured. The builder of the plants, the readers and the audit’s readers are agents of one model family, whose misses may be correlated (Knight and Leveson, 1986).
- *Resources.* 18 challenges are refused for a stack, memory or time limit that more resources might clear; 15 were not re-run by the operator’s choice, and the cap on the rented pod was set before the run. A refusal is not a rejection and is never counted as one.
- *One kernel upstream.* The upstream commit was decided with Lean’s kernel only; Comparator and the second kernel abort at start on this machine at that toolchain.

- *Selection.* The release was chosen by the author because it had a formal spine to decide; the results describe this release, not machine-written mathematics at large.

What is not claimed. Nothing about the truth of any theorem beyond the kernels’ acceptance of a proof of the statement the kernel saw; nothing about the mathematics of the manuscripts, whose weak points are recorded as notes; nothing about priority, whose dates are recorded as the parties state them; and nothing about the release’s authors, whose checks the gaps describe as what they do not show, never as errors.

6. Conclusions and recommendations

A machine-written release with a formal spine can be decided and read from its bytes, and the two answers are different in kind. The kernels accepted 398 of 416 challenges and the Navier–Stokes proof, and refused 18 for resources, each with its reason; the readings found 34 families whose formal statement says less than, or other than, its headline without saying so, and one clause of the Navier–Stokes theorem that was not formal at the announced commit; 14 gaps are pinned to file and line with the ask that would close each. To producers of formal releases: state beside every headline which of its clauses the formal statement carries, and where it is narrower, say so where the theorem is stated; switch the second kernel on, and ship the export files so that anyone can; declare the definitions a proof may leave as holes beside the theorem, not only in a configuration file; publish the finite witnesses the headlines rest on, with a digest beside each; and name the scope of every check in the release itself. To readers of such releases: a kernel’s acceptance is a fact about a statement the kernel saw, and the statement is what to read.

Declaration of competing interest

The author declares no competing financial or personal interests. OpenAI has not been consulted on this draft; issues and discussions are disabled on the release’s repository, and no message was sent to its authors. Prospective co-authors are reviewing this manuscript.

CRediT authorship contribution statement

Carlos Toledo: conceptualization, methodology, software, validation, investigation, data curation, writing—original draft, writing—review and editing. The author list and the roles are to be agreed with the prospective co-authors.

Data availability

Every record, lane and reading is public in the cert-machine repository (<https://github.com/carlostoledo1891/cert-machine>, commit 16849460e608; archived as Toledo, 2026): the pre-registration and its amendments `corpus/openai-math/preregistration.json`; the kernel lane `certs/openai-math-kernel.json` with every hosted run under `corpus/openai-math/kernel-runs/`; the statement lane `certs/openai-math-statements.json` with the readings under `corpus/openai-math/readings/`; the finite-core lane `corpus/openai-math/f-lane.json`; the gaps `corpus/openai-math/findings.json`; the recall measurement `certs/reading-recall.json` with its packets, keys and the readers' calls under `corpus/openai-math/recall-2026-10-19/`; the Navier–Stokes audit `corpus/navier-stokes/` (the manifest of pinned files, the build, the reading, the upstream decision); and the register row `certs/claims-ledger.json`. The release's bytes are the claimant's, cited at their source and pinned by SHA-256 digest; they remain under their owner's licence.

Declaration of generative AI and AI-assisted technologies

During the preparation of this work the author used large language models (Claude, Anthropic) to draft code, exposition and this manuscript in a certified-verification setting: constructions and prose were proposed with the help of the models, and every quantitative claim is decided by outward-rounded interval arithmetic or exact rational arithmetic, with adversarial controls that must fire for a record to be accepted. After using the tool the author reviewed and edited the content and takes full responsibility for the content of the publication. The records, not the models, are the evidence offered; they are public and re-runnable. For this paper specifically: the readings of the statements against their headlines, the five section memos and the Lean map of the Navier–Stokes manuscript, the census of finite objects and the planted defects of the recall measurement were produced by AI agents (Claude) under written briefs and consolidated into the public records; no verdict here rests on a reading alone, and no human specialist has confirmed the readings.

Funding

This work received no external funding. The rented compute (two CPU pods, the second within a \$10 cap) was paid by the author.

Appendix A. Reproduction

Every number in this paper is a macro written from a record of the repository at commit 16849460e608, and a record that no longer supports such a sentence stops the build. Each result re-derives with the command beside it in Table A.1; each command that fetches the claimant’s bytes refuses them unless they match the pinned digest.

result	command	expected	time
Navier–Stokes, the pin	the build that <code>corpus/navier-stokes/build.json</code> records, at 8937a8f4	both challenge theorems accepted by Comparator and the second kernel; only <code>propext</code> , <code>Classical.choice</code> , <code>Quot.sound</code>	140 min
Navier–Stokes, upstream	the incremental rebuild that <code>corpus/navier-stokes/upstream-f9e8bc5b.json</code> records, at f9e8bc5b	the energy clause a conclusion of <code>theorem_1_1</code> ; the same axioms for 7 declarations	25 min
the challenges, hosted	<code>.github/workflows/openai-math-kernel.yml</code> , one job per challenge	398 accepted by both kernels, 18 refused for resources, none rejected	18–12,453 s per run
the challenges, rented	<code>tools/runpod-openai-math.sh</code> on a CPU pod; <code>node tools/import-runpod-openai-math.js</code>	the same steps at the same pins; the facts served read-only	at the cap
the kernel record	<code>node tools/record-openai-math-kernel.js</code>	<code>certs/openai-math-kernel.json</code> re-derived from the run store	—
the statement record	<code>node tools/record-openai-math-statements.js</code>	<code>certs/openai-math-statements.json</code> re-derived from the readings	—
the finite objects	<code>python3 -I tools/run-openai-math-finite.py</code>	44 CERTIFIED, 7 REFUSED, 15 needing data; the red controls refuted	—
the readings’ recall	<code>python3 tools/score-reading-recall.py --check</code>	re-scored identical from the kept readings	—
this paper’s numbers	<code>node tools/paper-numbers/openai-math-release.js</code>	<code>paper/tex/openai-math-release-numbers.tex</code> rewritten identical	—

Table A.1: Each result, the command that re-derives it, what it must print, and what it took (the times are those the records hold;—where a record holds none).

References

- ammkrrn, 2026. `nanoda_lib`: a type checker for Lean 4 export files. URL: https://github.com/ammkrrn/nanoda_lib. Commit 3a240721, with the patch of issue #44.
- Avigad, J., 2024. Mathematics and the formal turn. *Bulletin of the American Mathematical Society* 61, 225–240. doi:10.1090/bull/1832.

- Bastounis, A., Circelli, F., Hansen, A.C., 2026. Navier–Stokes lost in translation: why Lean verification of AI autoformalisation does not guarantee correct natural language proofs. [arXiv:2610.08144](#).
- Carneiro, M., 2024. Lean4Lean: Verifying a typechecker for Lean, in Lean. [arXiv:2403.14064](#).
- Chen, J., Hou, T.Y., 2022. Stable nearly self-similar blowup of the 2D Boussinesq and 3D Euler equations with smooth data I: Analysis. [arXiv:2210.07191](#).
- Chen, J., Hou, T.Y., 2023. Stable nearly self-similar blowup of the 2D Boussinesq and 3D Euler equations with smooth data II: Rigorous numerics. [arXiv:2305.05660](#).
- Elgindi, T.M., 2021. Finite-time singularity formation for $C^{1,\alpha}$ solutions to the incompressible Euler equations on $\mathbb{R}^3$. *Annals of Mathematics* 194, 647–727. doi:10.4007/annals.2021.194.3.2.
- Fefferman, C.L., 2000. Existence and smoothness of the Navier–Stokes equation. URL: <https://www.claymath.org/millennium/navier-stokes-equation/>. The Clay Mathematics Institute’s official problem description.
- Google DeepMind, 2026. Formal conjectures: a collection of formalized conjectures in Lean. URL: <https://github.com/google-deepmind/formal-conjectures>.
- Hubert, T., Mehta, R., Sartran, L., et al., 2025. Olympiad-level formal mathematical reasoning with reinforcement learning. *Nature* doi:10.1038/s41586-025-09833-y.
- Kallel, M., et al., 2026. Verification abundance, adjudication scarcity. [arXiv:2608.28997](#).
- Knight, J.C., Leveson, N.G., 1986. An experimental evaluation of the assumption of independence in multiversion programming. *IEEE Transactions on Software Engineering* SE-12, 96–109. doi:10.1109/TSE.1986.6312924.
- Lean FRO, 2026. Comparator. URL: <https://github.com/leanprover/comparator>. Commit d03acab1.
- Lin, Y., Tang, S., Lyu, B., et al., 2025. Goedel-Prover: A frontier model for open-source automated theorem proving. [arXiv:2502.07640](#).
- Liu, Q., Zheng, X., Lu, X., Cao, Q., Yan, J., 2025. Rethinking and improving autoformalization: Towards a faithful metric and a

- dependency retrieval-based approach, in: The Thirteenth International Conference on Learning Representations (ICLR 2025). URL: https://proceedings.iclr.cc/paper_files/paper/2025/hash/d630537fc4402cfa3ebbc7450a0cac91-Abstract-Conference.html.
- Lu, J., Wan, Y., Huang, Y., et al., 2024. FormalAlign: Automated alignment evaluation for autoformalization. [arXiv:2410.10135](#).
- OpenAI, 2026a. math: manuscripts and Lean challenges. URL: <https://github.com/openai/math>. Commit fd4aeb2.
- OpenAI, 2026b. NavierStokesAndEuler. URL: <https://github.com/openai/NavierStokesAndEuler>. Commits 8937a8f4 and f9e8bc5b.
- Poiroux, A., Weiss, G., Kunčák, V., Bosselut, A., 2025. Reliable evaluation and benchmarks for statement autoformalization, in: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP 2025). [arXiv:2406.07222](#).
- Ren, Z.Z., Shao, Z., Song, J., et al., 2025. DeepSeek-Prover-V2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. [arXiv:2504.21801](#).
- Tao, T., 2016. Finite time blowup for an averaged three-dimensional Navier–Stokes equation. *Journal of the American Mathematical Society* 29, 601–674. doi:10.1090/jams/838.
- Toledo, C., 2026. cert-machine: independent exact certification of machine-generated mathematics. doi:10.5281/zenodo.22225860. The repository archive: code, records and reports. The concept DOI resolves to the latest version; the version this paper was built from is named on its title page.
- Wang, H., Unsal, M., Lin, X., et al., 2025. Kimina-Prover Preview: Towards large formal reasoning models with reinforcement learning. [arXiv:2504.11354](#).
- Wu, Y., Jiang, A.Q., Li, W., Rabe, M.N., Staats, C., Jamnik, M., Szegedy, C., 2022. Autoformalization with large language models, in: Advances in Neural Information Processing Systems 35 (NeurIPS 2022). [arXiv:2205.12615](#).