

Claude 5.5 models fall to half their 2024 knowledge of package releases 6 to 9 months before the stated cutoff and give the stated date in 63 of 70 answers inside Claude Code

7 models, 552 dated npm and PyPI releases, 200 versions that do not exist and a cutoff line the client adds to the system prompt

Dmitriy Semenkevich · [ORCID 0009-0009-3013-4978](https://orcid.org/0009-0009-3013-4978)

21 October 2026 · dimhold.by

ABSTRACT

A model's knowledge cutoff is stated by its vendor and repeated by the model itself. That the effective cutoff differs from the stated one is established. Dated Data (Cheng et al., 2024) showed it on open models through perplexity. LLM Lag Bench (Pezik et al., 2025) put the provider date, the model's own date and a measured boundary side by side for 35 models from news events. Hindsight Bench (Jia, 2026) found black box effective cutoffs on 15 models, up to 8 months before the vendor date for 5 of the 7 with a published date, Claude Sonnet 5 and Claude Haiku 4.5 among them. This study adds a narrower piece: the Claude 5.5 models, package releases as the subject and the self report as the Claude Code CLI delivers it. 552 releases from the npm and PyPI registries, 12 per month from 2023-01 to 2026-10, each dated by the registry itself, were asked about in batches of 20 with "do not know" allowed. 200 versions that do not exist were mixed into the same batches. The cutoff rule was registered before the first call. For the 4 models stated as Jun 2026 the last month at or above half of the 2024 level is 2025-09 to 2025-12, 6 to 9 months early. Correct answers reach 0 by 2026-02 or 2026-03. False recognition stays between 0 of 400 and 20 of 392, so the curve measures knowledge rather than guessing. The fall is a slope: on unsmoothed monthly shares it takes 6 to 9 months for 4 of the 6 models where the rule applies and 3 to 5 months for the other 2, where a step would read 1. Asked about their cutoff inside the CLI, the 4 models give Jun 2026 in 63 of 70 answers; all 7 give their vendor date in 118 of 130. The system prompt the CLI assembles carries "Assistant knowledge cutoff is June 2026" (captured for 1 model of 7). The answers are consistent with that line, but no condition without it was run. Asked with a prompt saying the line may be wrong, 25 of 25 answers from 5 models fall 3 to 10 months before their own curves.

1. The question

Every model card carries a knowledge cutoff, a month after which the model is not expected to know what happened. A programmer leans on that month every day without noticing. If the cutoff says June 2026, a suggestion to install a version released in March 2026 looks plausible and the absence of a feature from April looks like a gap in the library rather than in the model.

The month is stated by the vendor and, when asked, repeated by the model. Both are claims. This study measures the month by fact. It asks 7 models about 552 package releases whose publication dates the registries record to the second and reads off where the answers stop being right.

There are 3 numbers per model to put side by side: the vendor's, the model's own and the one the facts give. A fourth, the newest version the model can name per package, serves as an independent check on the third.

Throughout, **the 5.5 generation** means the 4 models stated as Jun 2026: `claude-opus-5-5`, `claude-sonnet-5-5`, `claude-haiku-5-5` and `claude-fable-5-1`, whose name does not carry the number 5.5.

2. Prior work

The question is not new and neither is the black box answer to it. 3 studies come before this one on the main axis.

Dated Data (Cheng et al., 2024, arXiv:2403.12958) introduced the effective cutoff, distinct from the stated one and showed that the 2 diverge. Its method is perplexity over dated versions of Wikipedia and CommonCrawl documents, which needs logits and open weights. It traces the drift to the data: old content reappearing in new CommonCrawl dumps and deduplication.

LLMLagBench (Pezik et al., 2025, arXiv:2511.12116) locates “the earliest probable temporal boundaries of an LLM’s training data” from questions about news events, with changepoint detection, on 35 models. Its Table 3 lists for each model the date according to

the developer, according to the model itself and according to the benchmark. That is the 3 numbers of this study, on a different subject and an earlier panel. Its Claude rows are Claude 3 Haiku, Claude 3 Opus, Claude 3.5 Sonnet and Claude Sonnet 4. For Claude Sonnet 4 it finds 2 changepoints, 2023-02 and 2024-12, against a provider date of 2025-01 and calls them “multiple partial cutoffs”.

HindsightBench (Jia, 2026, arXiv:2607.18867) is a preregistered black box audit protocol, without logprobs, on a macroeconomic panel. One of its metrics is the behaviorally effective cutoff, the last month at which the outcome recall propensity (LAP) stays above 0.1. LAP is how often, over 20 samples, the model offers a recalled outcome at all; whether the recall is right serves only as a gate. On 15 models from 7 vendors the cutoff is defined for 12 and spans 22 months. Where a vendor date is published, for 7 of the 12, it is early for 5, by up to 8 months, exact for 1 and 1 month late for 1. 2 of its models are in this panel. It measures Claude Sonnet 5 at 2025-10, 3 months before the reliable date of Jan 2026 and Claude Haiku 4.5 at 2024-10, 4 months before Feb 2025. Section 4.1 compares these with the numbers here.

Before them, **Beyond the Reported Cutoff** (Shah et al., 2025, arXiv:2504.00042) asked 197 thousand questions about the finances of US companies and compared the answers with fact by year, as a black box. Its resolution is a year. **llm-knowledge-cutoff** (Mazur, 2023) drew a monthly curve of share correct for 1 model, GPT-4 Turbo, from celebrity deaths, which is the method used here. **Set the Clock** (Zhao et al., 2024, arXiv:2402.16797) shows that pretrained models “mostly answer questions using earlier knowledge” than their pretraining cutoff.

On the shape of the fall, **TempoMed-Bench** (Guan et al., 2026, arXiv:2605.13045) finds on medical guidelines a gradual linear decline over time rather than a sharp edge at the cutoff. Section 4.3 reports the same kind of slope on package releases.

Nearby and not the same question: **Correct Code, Vulnerable Dependencies** (Wang et al., 2026, arXiv:2605.06279) records which library versions models write into code and finds 10 models converging on a small set, a consequence of the cutoff that it does not date. **LibEvoBench** (Cipollone et al., 2026, arXiv:2606.25402) tests whether models keep the APIs of several versions of Python libraries apart and finds them largely blind to the

version. **Simulated Ignorance Fails** (Li et al., 2026, arXiv:2601.13717) and **Can LLMs Be Constrained to the Past?** (Asai et al., 2026, arXiv:2606.05804) study whether a model can be made to forget what comes after a date. Public tables of stated cutoffs restate vendor dates without checking them.

How the search went. The search behind the registration ran on 2026-10-19, before the first model call, over Semantic Scholar (9 queries, raw responses in the repository), DuckDuckGo and GitHub. The arXiv API was throttled after 5 requests and returned nothing for a paper known to exist, so arXiv was in effect not searched. That search missed LLMagBench, HindsightBench, TempoMed-Bench and LibEvoBench. Set the Clock was among the Semantic Scholar responses and was not read. All 5 were found at review on 2026-10-21, after the counting. A search over arxiv.org/search on the same day (5 queries) found nothing closer.

What is left for this study is therefore narrower than the registration claimed:

- the 5.5 generation, which none of the 3 studies covers, plus `claude-opus-5` and `claude-sonnet-5` as the previous generation of the same lab on the same battery;
- package releases from npm and PyPI as the subject, dated by the registries to the second and asked at monthly resolution;
- an explicit control of 200 versions that do not exist, asked in the same batches as the real ones;
- the self report as a commercial client delivers it: inside the Claude Code CLI, whose system prompt carries a cutoff line of its own. LLMagBench reports self dates earlier than the provider's for all 4 Claude models it lists (for Claude Sonnet 4, 2024-04 against 2025-01). Inside the CLI the 5.5 generation gives the vendor date instead.

3. Method

3.1 The battery

`battery.py` pulls the full release history of 100 popular package entries, 50 from npm and 50 from PyPI, from the registries on 2026-10-19. For each minor line X.Y it keeps the first stable release, usually X.Y.0. Patch versions are left out because no model is expected

to remember them and prereleases are left out as well. `nestjs` on npm and `airflow` on PyPI hold only placeholders (0.0.1 and 0.6) and have no stable release in the window. The real packages live under `@nestjs/core` and `apache-airflow`, which are in the list too.

Within each month from 2023-01 to 2026-10 the script takes up to 12 releases, at most 1 per package before any package gets a second, shuffled with a fixed seed. 46 months at 12 gives **552 real releases**. They come from 93 of the 98 entries that have a stable release in the window; the other 5 lost every draw to the monthly cap. The date of each release is the registry's own record of first publication. Nothing in the reference is written from memory.

Beside them stand **200 versions that do not exist**: the next major and the next minor after the latest published version of each of the 100 entries, including the 2 placeholders, each checked against the registry's full version list on the snapshot day. These measure how often a model claims to know something that is not there.

3.2 The measures and the cutoff rule

`CRITERIA.md` was written before the first call and registers the following.

Knowledge curve. For each real release the model says whether it knows that the version was published and, if so, the month. An answer is correct when `known` is true and the month is at most 1 away from the registry. The share correct is computed per month of publication over 2 repeats, so each cell holds 24 answers per model.

Measured cutoff. The last month after which the 3 month rolling share correct falls below half of the model's mean share for 2024 and never rises above it again. In practice the rule returns the last month whose window is still at or above half of that mean.

False recognition. The share of `known` set to true on the 200 versions that do not exist, over 2 repeats. If it were above 20%, the registration said the curve would be measuring a tendency to guess and would be replaced.

Frontier. For each package, the most recent stable version the model can name. The named version is looked up in the registry and its publication month is the model's frontier for that package. Median over packages, 2 repeats.

Self report. 6 wordings of “what is your knowledge cutoff”, 5 repeats each.

Added after registration. The registration has no interval and no percentile. The bootstrap of the measured cutoff (2000 resamples drawn within each month, the rule applied again to each), the Wilson intervals, the 90th percentile of the frontier, the monthly slices, the width of the decline in section 4.3 and the 10% threshold of the comparison with HindsightBench in section 4.1 were all chosen after the data came in. The 90th percentile is reported because the maximum of the frontier turned out unreliable (section 4.1). So were the 2 arms `self2` and `guess0` (section 3.4).

3.3 The harness

The questions go to the models through the Claude Code CLI 2.1.302 on a subscription:

```
claude -p --output-format json --model <model> \  
  --system-prompt "You are a helpful assistant." --tools "" \  
  --strict-mcp-config --mcp-config no-mcp.json < prompt.txt
```

The panel is the 5.5 generation, all stated as Jun 2026, then `claude-opus-5` (May 2026), `claude-sonnet-5` (Jan 2026) and `claude-haiku-4-5` (reliable knowledge cutoff Feb 2025, training data cutoff Jul 2025). The vendor dates are taken from Anthropic’s model pages as captured on 2026-10-19: an excerpt of the overview page and the 3 legacy model pages are in the repository.

The run directory is empty and every parent directory up to the root is checked for `CLAUDE.md`, `.claude`, `AGENTS.md` and `.cursorrules`, because the CLI loads such files into the prompt without saying so. The model that answered is read from the `modelUsage` keys of each response envelope. In 1075 calls of 1075 the requested model was the only key.

Questions are asked in **batches of 20**. Real and nonexistent versions are shuffled together and a batch is identical for every model, so false recognition is measured in exactly the condition knowledge is. Batching was a decision forced by time: single calls in the `self` phase took 35 to 132 s, median 64 and 10000 of them did not fit into the window.

The CLI adds context of its own. The first calls of the run were a context check asking 2 models to quote everything in front of the prompt (`context-check.json`). `claude-haiku-5-5` quoted the system prompt in full. `--system-prompt` does not replace it: the text passed with it comes after a line the CLI keeps, “You are a Claude agent, built on Anthropic’s Claude Agent SDK.”. After that comes an environment block with the working directory, platform, OS version, shell and a paragraph about untrusted files. Then a model identity line, “You are powered by the model named Haiku 5.5. The exact model ID is `claude-haiku-5-5`.”, followed in the same paragraph by “Assistant knowledge cutoff is June 2026.”. Then a token budget line and “Today’s date is 2026-10-19”. A system reminder with the account email is attached to the user message. `claude-opus-5-5` quoted the first line, the passed system prompt and the reminder. It said that “there is also an environment block, but it appears after your message”, so the block is there too, though its content was not quoted. For the other 5 models the context was not captured.

The registration knew about the date line. It records that the date and the working directory survive `--system-prompt` and `--exclude-dynamic-system-prompt-sections` and that `--bare` does not log in on a subscription. Whether those flags remove the cutoff line was not tested: the line was found only by this context check and the run itself used `--system-prompt` alone.

3.4 What was taken

PHASE	WHAT	SUCCESSFUL CALLS
<code>batch0</code> , <code>batch1</code>	knowledge curve and false recognition, 2 repeats	266 + 266
<code>front0</code> , <code>front1</code>	frontier, 2 repeats	56 + 56
<code>self</code>	self report, 6 wordings	130 of 210
<code>self2</code>	self report with a prompt saying the cutoff line may be wrong	35 of 35
<code>guess0</code>	month estimate required, 1 repeat	266

The journal holds 1243 rows. **168 failed rows are excluded:** 163 where the OAuth session expired overnight and could not be refreshed and 5 where the CLI received no prompt on stdin within 3 s. Every one of those cells was taken again successfully later and no successful key appears twice. 3 batches of 532 did not parse completely, 1 for `claude-sonnet-5` and 2 for `claude-haiku-4-5`. The nominal cost recorded by the CLI is \$36.96 for the 1075 successful calls.

`self2` and `guess0` were not in the registration. Both were added during the run and section 4 reports them as additions.

4. Results

4.1 The curve falls 6 to 9 months early

MODEL	STATED	MEASURED	95% BOOTSTRAP	MONTHS EARLY	FRONTIER MEDIAN	FRONTIER P90
claude- opus-5-5	2026-06	2025-09	2025- 08..2025-10	9	2025-08	2025-10
claude- sonnet-5- 5	2026-06	2025-12	2025- 12..2026-01	6	2025-10	2025-12
claude- haiku-5- 5	2026-06	2025-09	2025- 08..2025-10	9	2025-09	2025-10
claude- fable-5- 1	2026-06	2025-11	2025- 10..2025-11	7	2025-09	2025-11
claude- opus-5	2026-05	2025-10	2025- 09..2025-10	7	2025-10	2026-02
claude- sonnet-5	2026-01	2025-01	2024- 12..2025-02	12	2025-02	2025-09
claude- haiku-4- 5	2025-02	not applicable			2024-02	2024-07

For all 4 models of the 5.5 generation the last month at or above half of the 2024 level falls between 2025-09 and 2025-12. In that month they still get 10 to 17 of 24 releases right. The upper bootstrap bound is 2026-01 at the latest, so the gap is at least 5 months for every one of them. The registered expectation was a gap of at least 2 months. It holds with room to spare and the refutation, a match to the month for every current model, fired for none.

What that looks like month by month, for `claude-opus-5-5` (correct answers of 24): 22 in 2025-01, 17 in 2025-06, 11 in 2025-08, 3 in 2025-12 and 0 from 2026-02 onward. The 5.5 generation reaches 0 correct in 2026-02 (2026-03 for `claude-sonnet-5-5`), 3 to 4 months before Jun 2026. From 2026-03 to 2026-09 no model of the 7 gives a single correct answer. The one exception after that is 4% for `claude-sonnet-5` in 2026Q4.

The measured cutoff is stable between repeats. Repeat 0 and repeat 1 differ by at most 2 months, for `claude-opus-5-5` (2025-10 against 2025-08) and coincide for 4 of the 7 models.

The frontier agrees without sharing a question. It asks for the newest version of each package rather than about given versions and its median for the 5.5 generation lies in 2025-08..2025-10. The maximum is not reported as a frontier because it is unreliable: for `claude-opus-5` it reaches 2026-08 in a set where 13 of the 200 named versions do not exist.

The newer generation does not know further than the older one. `claude-opus-5-5`, stated a month later than `claude-opus-5`, measures at 2025-09 against 2025-10 with overlapping intervals. `claude-sonnet-5` diverges most of all: stated Jan 2026, measured 2025-01, frontier median 2025-02. Its curve already sags in 2024, with a 2024 baseline of 0.719 against 0.812 to 1.000 for the 5.5 models and 0.972 for `claude-opus-5`.

Against HindsightBench. The 2 studies share `claude-sonnet-5` and `claude-haiku-4-5`. HindsightBench thresholds a recall propensity, not a share correct, so the closest measure here is the share of answers with `known` set to true. Both shares are read below at the same 10% threshold, which was chosen after the data. On its macroeconomic panel HindsightBench measures Claude Sonnet 5 at 2025-10. Here the last month with more than 10% `known` is 2025-06 (5 of 24), the share correct gives the same month and the registered rule gives 2025-01, so the 2 studies disagree by 4 to 9 months and both put the model well before its vendor date of Jan 2026. For Claude Haiku 4.5 it measures 2024-10. Here the rule cannot be applied. The last month with more than 10% `known` is 2024-08 (3 of 24), 2 months earlier and the estimate arm shows hits up to 2025-02 (section 4.4). Against the reliable date of Feb 2025 the 2 studies differ: HindsightBench puts the model 4 months before it and this study finds no knowledge beyond it, which is not the same as

before it. Both are before the training date of Jul 2025. The gap on `claude-sonnet-5` fits the Dated Data point that the effective cutoff depends on the resource, but 2 models are too few to say more.

4.2 False recognition is low enough to read the curve

MODEL	KNOWN ON NONEXISTENT VERSIONS	95% WILSON INTERVAL
<code>claude-opus-5-5</code>	0/400	0.0..1.0%
<code>claude-sonnet-5-5</code>	0/400	0.0..1.0%
<code>claude-haiku-5-5</code>	3/400	0.3..2.2%
<code>claude-fable-5-1</code>	1/400	0.0..1.4%
<code>claude-opus-5</code>	4/400	0.4..2.5%
<code>claude-sonnet-5</code>	2/394	0.1..1.8%
<code>claude-haiku-4-5</code>	20/392	3.3..7.7%

The registered threshold was 20%. The highest rate is 5.1%. The curve measures knowledge and the primary measure stands.

Confidently wrong answers on real releases tell the same story for most of the panel. For `claude-opus-5-5`, `claude-sonnet-5-5`, `claude-fable-5-1` and `claude-opus-5` they run from 0 to 4% per quarter and from 0 to 1% after 2023Q1. `claude-haiku-5-5` is the exception, at 10 to 32% per quarter up to 2025Q4: it is the one model of the 5.5 generation that claims to know a release and names the wrong month. In the previous generation `claude-sonnet-5` reaches 13% and `claude-haiku-4-5` 12% in single quarters of 2023.

4.3 The cutoff is a slope

The registration named a refutation: a smooth decline from 2023 with no break, in which case a cutoff as a point would not exist. It did not fire. For `claude-opus-5-5`, `claude-sonnet-5-5`, `claude-fable-5-1` and `claude-opus-5` the share correct sits on a plateau

of 81 to 100% per quarter from 2023Q1 to 2025Q2, then falls. `claude-haiku-5-5` has a lower plateau (65 to 89%) and `claude-sonnet-5` starts falling earlier, in 2025Q1.

The fall takes time, though. The width below is counted from the last month at 90% or more of the 2024 baseline to the first month after it at 10% or less. It is given twice: on the 3 month rolling window the rule uses and on the unsmoothed monthly share. The window has a floor. A perfect step, all correct through month m and none after, reads 1 at m-1, 2/3 at m, 1/3 at m+1 and 0 at m+2, so its width is 3 months. On the unsmoothed share the floor is 1 month.

MODEL	SMOOTHED: LAST AT 90% OR MORE	FIRST AT 10% OR LESS	MONTHS	UNSMOOTHED MONTHS
<code>claude-sonnet-5-5</code>	2025-11	2026-03	4	4
<code>claude-opus-5</code>	2025-07	2026-01	6	3
<code>claude-fable-5-1</code>	2025-07	2026-02	7	7
<code>claude-haiku-5-5</code>	2025-05	2026-01	8	6
<code>claude-opus-5-5</code>	2025-05	2026-02	9	9
<code>claude-sonnet-5</code>	2024-11	2025-08	9	9

The unsmoothed width of `claude-opus-5` sits on a tie. 21 of 24 in 2025-09 is exactly 90% of its baseline. A strict comparison would make the width 5.

On the unsmoothed share, where a step would read 1 month, every model takes 3 months or more. For `claude-opus-5-5`, `claude-sonnet-5`, `claude-fable-5-1` and `claude-haiku-5-5` the fall takes 6 to 9 months. `claude-sonnet-5-5` and `claude-opus-5` are the narrowest, at 4 and 3 months (5 for `claude-opus-5` under a strict comparison). They still ramp rather than drop: 21, 17, 13, 3 and 0 correct of 24 over 2025-11..2026-03 for `claude-`

`sonnet-5-5` and 21, 11, 6 and 2 over 2025-09..2025-12 for `claude-opus-5`. The 1 month tolerance in what counts as correct widens every band a little. The point produced by the registered rule lies inside the band.

A gradual fall rather than a sharp edge is what TempoMed-Bench reports for medical knowledge. LLMlagBench describes a different shape: it fits piecewise constant segments and for Claude Sonnet 4 finds 2 changepoints, a step down to a lower plateau at 2023-02 and a sharp drop at 2024-12. That is more than 1 boundary per model, but a set of steps rather than a slope. Why the band sits months before the stated date cannot be read from a black box. Dated Data traced the same kind of drift in open models to the composition of the training data.

4.4 `claude-haiku-4-5`: the rule does not apply

The registration expected `claude-haiku-4-5` to know beyond its reliable cutoff of Feb 2025 but not beyond its training cutoff of Jul 2025. In the primary arm it almost always says it does not know. For 2024 it gives 12 correct answers of 275 pooled; the baseline, which the rule defines as the mean of the monthly shares, is 4.3%. Half of 4.3% is not a threshold anything can be read from and the rule’s output for this model (2024-12, bootstrap 2024-04..2025-01) is reported only for completeness.

The model can be read only from the arm in which a month estimate is required. That arm was not registered, ran once and, as section 4.6 shows, does not measure knowledge in general. For this model it is still informative, because its hit rate on old releases stands well above its hit rate on releases it cannot have seen:

WINDOW	ESTIMATE WITHIN 1 MONTH	95% WILSON INTERVAL
2024-01..2024-12	50/144 = 34.7%	27.4..42.8%
2025-01..2025-02	9/24 = 37.5%	21.2..57.3%
2025-03..2025-07	2/60 = 3.3%	0.9..11.4%
2025-08..2026-10	5/180 = 2.8%	1.2..6.3%

These hits are almost all answers the model itself does not call memory. It flags “remember” for 0 to 8% of items per quarter. Among its “do not remember” answers from 2023-01 to 2025-06 the month still lands within 1 in 27%. So this is a signal visible in unflagged estimates, not knowledge the model reports. That signal holds to 2025-02 and then drops to the level of months after the training cutoff. The 2025-01..2025-02 window is only 24 answers, with an interval of 21 to 57%. The window between the reliable and the training cutoff is indistinguishable from the period after both. The frontier agrees: maximum 2025-02, p90 2024-07. The first half of the expectation is not confirmed. With sensitivity this low, the finding reads “no knowledge is visible beyond Feb 2025” rather than “there is none”.

4.5 The self report gives the vendor date

The registration expected the self report to come in earlier than both the stated and the measured cutoff. In the condition the CLI actually runs in, it came in later than the measured one. The 5.5 generation names Jun 2026 in **63 of 70 answers** and all 7 models name exactly their vendor date in **118 of 130**:

MODEL	ANSWERS	SELF REPORT IN THE CLI	SELF2, TOLD THE LINE MAY BE WRONG
claude-opus-5-5	18	2026-06 x18	2025-01 x5
claude-sonnet-5-5	19	2026-06 x18, 2026-05 x1	2025-05 x2, 2025-06, 2025-08, 2025-09
claude-haiku-5-5	13	2026-06 x12, 2026-05 x1	2025-01 x2, 2025-03 x2, 2025-05
claude-fable-5-1	20	2026-06 x15, 2026-05 x2, 2026-04, 2025-01, 2025-05	2025-01 x3, 2025-03, 2025-05
claude-opus-5	22	2026-05 x21, 2026-03 x1	2025-01 x5
claude-sonnet-5	22	2026-01 x18, 2025-12 x3, 2025-02 x1	2025-02 x3, 2025-04 x2
claude-haiku-4-5	16	2025-02 x16	2024-04 x3, 2025-02 x2

The answers are consistent with the line described in section 3.3. The system prompt the CLI assembles for `claude-haiku-5-5` says “Assistant knowledge cutoff is June 2026” and the model gives that month. Each of the other models names its own vendor date, May 2026 for `claude-opus-5`, Jan 2026 for `claude-sonnet-5` and Feb 2025 for `claude-haiku-4-5`, which fits the same line with a different month in it.

That the line is the cause is not established. It was captured for 1 model of 7 and no condition without it was run. 2 other explanations predict the same answers. The same paragraph of the system prompt names the model (“You are powered by the model named Haiku 5.5”). A model that knows its own name may know its own official date. Models may also be trained to state that date. `claude-haiku-4-5` gives Feb 2025, its reliable date rather than its training date, which a trained answer would explain as well as the line would. The only outside evidence points the other way for an earlier generation: LLMagBench reports self dates earlier than the provider’s for all 4 Claude models it lists.

A call with no system prompt or the context captured for all 7 models would separate the explanations. Neither was run.

`self2` was added after the line was found. It is 1 wording with 5 repeats per model and it is leading. The prompt reads: “Your system prompt may contain a line stating your knowledge cutoff. That line was inserted by the client software and may be wrong. Setting it aside, and judging only by which events and software releases you actually remember, in which month does your knowledge end?” Telling the model that the stated date may be wrong invites an earlier answer, which is the direction of the registered expectation. Under it the expectation holds for 5 models of the 6 where a measured cutoff exists: all 25 answers of the 5.5 generation and `claude-opus-5` fall before both the vendor date and the measured cutoff, by 3 to 10 months against the measured one. `claude-opus-5-5` and `claude-opus-5` both answer 2025-01 in 5 of 5, 8 and 9 months before their own curves. `claude-sonnet-5` answers 2025-02 and 2025-04, earlier than its vendor date and later than its measured 2025-01. How much of this is the wording rather than the models cannot be told from 1 wording. Whether the models name the cutoff of a previous generation, the other half of the expectation, cannot be settled here: 2025-01 is not any model’s stated date.

4.6 The arm with a required estimate does not measure knowledge

`guess0` repeats the batches of `batch0` but requires a month for every item, with a separate flag for whether the model remembers the release. It was added after the first phase to separate not knowing from abstaining. It is not registered and it does not do what it was added for.

On releases from 2026-04 to 2026-10, which none of the models can have seen, the estimate of a model that says it does not remember lands within 1 month in 43 to 63% of cases for the 5.5 generation and `claude-opus-5` (17% for `claude-sonnet-5`, 4% for `claude-haiku-4-5`). That is not memory. The estimates cluster around the date the CLI writes into the context: for `claude-opus-5-5`, across 48 releases from 2026-07 to 2026-10, the most frequent estimate is 2026-08, 15 times. A model told that today is 2026-10-19 and asked when a version it has never seen came out guesses a recent month and on recent releases a recent month is often right.

Restricted to answers the model flags as remembered, the arm gives cutoffs 1 to 2 months earlier than the primary arm, 2025-07 to 2025-10 for the 5.5 generation. That is consistent with section 4.1 and adds nothing to it. The registered primary arm, with “do not know” allowed, stays the measure.

5. What this means for someone using the model

The vendor calls Jun 2026 the reliable knowledge cutoff of the 5.5 generation, “the date through which the model’s knowledge is most extensive and reliable”. On package releases it is not. By late 2025 these models know about half of what they knew of 2024. The months after that are a slope on which the model knows some releases and not others. A version released from 2026-02 on is almost always unknown to them: the best of the 7 knows 3 of 24 releases from that month. A model asked inside Claude Code what its cutoff is gives the vendor date, so that answer is not evidence either way.

6. Threats to validity

The cutoff line in the system prompt. The CLI adds a cutoff line to the system prompt, next to today’s date and the model’s name. Of the flags tried, none removes the date line. Whether they remove the cutoff line was not tested (section 3.3). The line was observed for `claude-haiku-5-5` only and is inferred for the other 6 models from their answers. In the primary arm it is the same for every question and false recognition of 0 to 1% for the current models suggests it does not push them to invent releases. Its effect on how often a model says “known” was not measured. The self report in the working condition gives the vendor date. Whether because of the line, the model name next to it or training is not separated (section 4.5).

The `self2` wording is leading. It tells the model the line may be wrong, which pushes toward an earlier answer. It is 1 wording with 5 repeats and its 25 of 25 cannot be split into what the models know and what the prompt suggests.

The arm with a required estimate. It was not registered, it ran once and its estimates cluster around the date the CLI supplies. It is used only to read `claude-haiku-4-5`, where

the primary rule cannot be applied and that verdict rests on an arm outside the registration, on answers the model does not flag as remembered.

The self phase was cut. It stopped at 130 of 210 calls when the run moved to batches, leaving 13 to 22 answers per model. Which cells were taken depends on the order the harness ran them in, so it is not a random subsample in the strict sense.

claude-haiku-4-5 and the rule. With a 2024 baseline of 4.3% (12 of 275 pooled) the registered rule produces a month that means nothing. Its row in the main table is left without a measured cutoff for that reason.

The width of the slope depends on the smoothing. The rule's 3 month window cannot read a step as narrower than 3 months. The unsmoothed widths are given beside it. Their floor is 1 month and all 6 are 3 months or more.

A narrow subject. The battery is months of minor releases of 93 popular packages. This is a cutoff for package versions, not for knowledge in general and popular packages are probably better represented in training data than the average fact. HindsightBench measures **claude-sonnet-5** 4 to 9 months later on a different subject.

Batches of 20. Neighbouring versions of 1 package in the same batch can suggest a month by interpolation and the 1 month tolerance makes that easier. False recognition was measured in the same batches and is low, but low false recognition does not rule out interpolating the month of a version that does exist.

Small cells. 2 repeats for the primary arm and the frontier, 1 for the estimate arm. Each month holds 12 releases, so each cell of the curve is 24 answers per model.

A subscription, not the API. Temperature and other sampling parameters are whatever the CLI uses and are not under control. The answering model was verified on every call. Its sampling settings were not.

The registration has no third party timestamp. It is dated by its git commit, 16 minutes before the context check and 18 minutes before the first journal row. The public history was rebuilt for publication (translation and redaction), so that order rests on local git dates set by the author.

The prior work search missed the closest studies. The search behind the registration did not reach arXiv and missed LLM LagBench, HindsightBench and TempoMed-Bench (section 2). The novelty claim in the registration was wider than what is left after them.

7. Availability

The battery and the script that builds it, the full registry extract it was built from, the harness, both analysis scripts, the context check, the registration, the prior work search with raw Semantic Scholar responses, an excerpt of the vendor overview page with the 3 legacy model pages and the journal of all 1243 calls with every prompt and reply are in the repository. Every table and every number quoted from the journal in this paper is printed by `analyze.py` or `intervals.py`, offline and seeded. The account email and local paths were redacted before publication and the redaction is listed in the README.

Archived on Zenodo with DOI [10.5281/zenodo.23544586](https://doi.org/10.5281/zenodo.23544586). Repository: github.com/dimhold/cutoff-probe.

8. References

1. Jeffrey Cheng, Marc Marone, Orion Weller, Dawn Lawrie, Daniel Khashabi, Benjamin Van Durme. Dated Data: tracing knowledge cutoffs in large language models. arXiv preprint, 2024. arXiv:2403.12958. <https://arxiv.org/abs/2403.12958>
2. Piotr Pęzik, Konrad Kaczyński, Maria Szymańska, Filip Żarnecki, Zuzanna Deckert, Jakub Kwiatkowski, Wojciech Janowski. LLM LagBench: identifying temporal training boundaries in large language models. arXiv preprint, 2025. arXiv:2511.12116. <https://arxiv.org/abs/2511.12116>
3. Haozhe Jia. HindsightBench: a black-box behavioral audit protocol for parametric hindsight in time-indexed LLM decision tasks. arXiv preprint, 2026. arXiv:2607.18867. <https://arxiv.org/abs/2607.18867>
4. Agam Shah, Liqin Ye, Sebastian Jaskowski, Wei Xu, Sudheer Chava. Beyond the reported cutoff: where large language models fall short on financial knowledge. arXiv preprint, 2025. arXiv:2504.00042. <https://arxiv.org/abs/2504.00042>

5. Matt Mazur. llm-knowledge-cutoff: a notebook to evaluate the accuracy of an LLM's knowledge over time. GitHub, 2023. <https://github.com/mattm/llm-knowledge-cutoff>
6. Bowen Zhao, Zander Brumbaugh, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith. Set the Clock: temporal alignment of pretrained language models. arXiv preprint, 2024. arXiv:2402.16797. <https://arxiv.org/abs/2402.16797>
7. Zihan Guan, Qiao Jin, Guangzhi Xiong, Fangyuan Chen, Mengxuan Hu, Qingyu Chen, Yifan Peng, Zhiyong Lu, Anil Vullikanti. Large language models lack temporal awareness of medical knowledge. arXiv preprint, 2026. arXiv:2605.13045. <https://arxiv.org/abs/2605.13045>
8. Chengjie Wang, Jingzheng Wu, Xiang Ling, Tianyue Luo, Chen Zhao. Correct code, vulnerable dependencies: a large scale measurement study of LLM-specified library versions. arXiv preprint, 2026. arXiv:2605.06279. <https://arxiv.org/abs/2605.06279>
9. Daniele Cipollone, Sergey Titov, Maliheh Izadi, Egor Bogomolov, Arie van Deursen. LibEvoBench: probing temporal knowledge stratification in code generation models. arXiv preprint, 2026. arXiv:2606.25402. <https://arxiv.org/abs/2606.25402>
10. Zehan Li, Yuxuan Wang, Ali El Lahib, Ying-Jieh Xia, Xinyu Pi. Simulated ignorance fails: a systematic study of LLM behaviors on forecasting problems before model knowledge cutoff. arXiv preprint, 2026. arXiv:2601.13717. <https://arxiv.org/abs/2601.13717>
11. Michiro Asai, Ailiang Lin, Yu Kishimoto, Takao Obi, Satoshi Kosugi, Kotaro Funakoshi, Manabu Okumura. Can LLMs be constrained to the past? Improving knowledge cutoff through recall-based prompting. arXiv preprint, 2026. arXiv:2606.05804. <https://arxiv.org/abs/2606.05804>
12. Dmitriy Semenkevich. `cutoff-probe`: battery, harness, registration and the journal of every call. GitHub, 2026. <https://github.com/dimhold/cutoff-probe>

CITE THIS

Semenkevich, D. (2026). Claude 5.5 models fall to half their 2024 knowledge of package releases 6 to 9 months before the stated cutoff and give the stated date in 63 of 70 answers inside Claude Code. dimhold.by. <https://doi.org/10.5281/zenodo.23544586>

LLM evaluation · knowledge cutoff · temporal knowledge · software package versions · model self report · black box probing · preregistration