Local-computation optimization: before / after study

green = saved ≥20% · red = cost ≥20% more · black = within ±20% = direction-at-best. The ±20% band is the noise floor this doc demonstrates (cross-session drift ~8pp + N=1 task variance), so within-band magnitudes are not coloured as signal. Colours the "% saved" columns only; swing/pp and token columns stay neutral.

Date: 2026-05-31. Methodology: five rounds of paired A/B testing across three Claude Code skills, with the model held constant per cell so the delta controls for background model drift. (It does not by itself isolate the optimization, the within-round delta also carries the skill's fixed loading overhead; see "What this measures".)

Companion PDF: `skills-optim-study-2026-05-31.pdf`. Raw data: `skills-optim-study-2026-05-31.json` (Round 1), `-round2.json`, `-rounds-3-4-5.json`.

tl;dr

The one durable result. Bounded, scoped procedural language in a SKILL.md reliably cuts token use for **Haiku-class models**; for **Opus-class** the effect sits inside the measurement noise. There is **no portfolio-level save claim** in this data: three skills, N=1 per cell.

> **Update (round 6 (2026-06-01).** The "inside the noise" read for Opus above is about the N=1 *swing* (before-fix minus after-fix) a difference of two fragile ratios). A later depth re-measurement measured the Opus *level* (skill-vs-cold) instead, at N=5/arm with both arms stable, and found a real **+76%** save for both auditor skills: at depth the cold arm does the full audit while the script-backed skill short-circuits, and the earlier near-zero/negative reads were N=1 artifacts of an unusually shallow cold arm. The fragile N=1 *swings* were noise; the underlying Opus save at depth is real. Read what follows as the rounds-1–5 record: the depth correction lives in the replicates.

The strongest single piece of evidence is a swing, not an aggregate. skills-freshness on Haiku went **−70%** → **+20%** (a **+90pp** swing) once a `limit=80` / "don't spelunk" guard landed in the SKILL.md (PR #18). Big effect, an identified mechanism (a 233K-char full-file read), a targeted fix, and a confirmed reversal: large enough to clear the N=1 noise. The other large swing (content-audit/haiku **+16pp**, R3→R4) repeats the pattern. The transferable output is the **three cost-trap mechanisms** below; the percentages are the evidence that they are real.

Two questions live in this study, and they want different numbers:

- *Did the optimization actually work?* → read the **swing** (R1→R2, R3→R4). Across rounds the cold arm holds the same role (no skill) while the SKILL.md content is what changes, so the swing **targets** the optimization, but because the cold arm is re-run rather than pinned, it also carries run-to-run variance, which is why only the large swings are trustworthy (see Method → Pin the BEFORE arm).
- *Is the skill, in its current state, cheaper than no skill at all?* → read the **within-round aggregate** (R1, R5). This is close to what the calibration already answered, and it bundles the fixed cost of loading a SKILL.md + dispatching a script into the AFTER arm.

The round-by-round aggregates below are the **second** number. They are honest, but they are **not** the optimization's effect: treat them as supporting context and read the swings + mechanisms as the result.

Round	Skills measured	Cells	Agg. (skill vs cold)	What it showed
1	skills-freshness, skills-quality	6	+2.5%	3 of 6 cells sign-flipped, surfaced the cost-trap mechanisms

Round	Skills measured	Cells	Agg. (skill vs cold)	What it showed
2	skills-freshness (post-fix)	3	**+11%**	all 3 cells turned around; freshness/haiku **+90pp** swing
3	content-audit (rigor-exempt test)	3	+1%	rigor-exempt label inflated, sat at break-even
4	content-audit (post-fix)	3	+4%	MIXED, haiku **+16pp** flip; sonnet −6pp swing (within noise)
5	both new skills (re-validation)	6	+16%	MIXED, strongest aggregate, but concentrated in two haiku cells

42 sub-agents total, ~4.48M subagent tokens spent measuring. **N=1 per cell**: sufficient to surface direction, not magnitude.

The three cost-trap mechanisms (the transferable result)

This is the part worth keeping. Round 1's forensic pass found three concrete ways a procedural SKILL.md turns *against* token economy: each visible in the JSONL transcripts, each with a guard wording that neutralises it. Each was observed on a single cell here, so the *frequency* is unmeasured, but the *susceptibility* is structural: any SKILL.md written in imperative prose can trigger all three.

#	Mechanism	How the prose triggered it	Guard wording that fixes it	PR #18 rule
1	**Unlimited read**	"read each SKILL.md" → the AFTER arm read **233,876 chars** across 14 files in full vs the cold arm's 116,731	`limit=80`, frontmatter + first one or two sections is all you need; no deep read	`unlimited_read_in_procedure`
2	**Uncapped follow-up / spelunking**	a missing cap let the agent chase broken path refs into the source repo: +4 Bash calls, **+28K tokens**	"one `ls` per finding, max 3 traces total, don't spelunk into source repos"	`uncapped_followup`
3	**Batch invitation**	"read in parallel" was staged as a 6+8-file batch, paying **+27K cache_creation tokens**	cap the batch; don't pre-stage a large parallel read on the model's behalf	`batch_invitation`

The char and token figures in the table are forensic readings from Round 1's per-agent JSONL transcripts, not fields in the summary JSON raw-data files. The common root: each fired because the SKILL.md procedure **removed the scoping pressure** the improvised (cold) arm still felt. An unguided model rations its own reads when it is paying for them; a procedure that says "read each X" hands it permission to stop rationing. The guard wording puts the ceiling back. **This taxonomy is the reusable artifact of the study**: the percentages below exist to show it is real, and to show the fixes land.

What this measures, and what it does not

The 2026-05-22 Spacepotatis skills calibration measured a ~22% net savings rate across 13 skills, replacing the editorial 3× heuristic with real numbers. That answered "how much do my skills save?": a portfolio question.

This study set out to answer a *different* one: **for a specific optimization, did it actually work?** But "the optimization" is two things, and the design measures them with two different numbers:

- **Creating the skill** (`claude-skills` PRs #14–#17 created skills-freshness and skills-quality). For a brand-new skill the only available "before" is *no skill*, so the within-round aggregate (AFTER-with-SKILL.md vs BEFORE-no-SKILL.md) is the right comparison, but it is **skill-vs-cold**, and it bundles in the fixed overhead of loading the SKILL.md and dispatching its script. That overhead is exactly why 3 of 6 Round-1 cells went negative.
- **Tightening the skill** (PR #18 / Spacepotatis #280 added the guard wording above). Here the cold arm is held fixed in role and the SKILL.md content is the only *intended* change between rounds, so the **swing** (R1→R2, R3→R4) **targets** the tightening's effect, though the re-run cold arm still adds variance (see Method → Pin the BEFORE arm).

Read in that light: the within-round aggregates partly re-answer the calibration's question; the **swings** are the part that is genuinely new here. The headline belongs to the swings and the mechanisms, not the aggregate.

Round 1: original measurement (2 skills × 2 arms × 3 models)

Hypothesis: the two new skills (`skills-freshness`, `skills-quality`) should net-save tokens vs unguided improvisation.

Cell	BEFORE	AFTER	Saved	%	Verdict
skills-freshness/sonnet	194,751	110,828	+83,923	+43%	strongest save
skills-freshness/opus	134,765	163,301	−28,536	−21%	sign-flip
skills-freshness/haiku	64,192	108,806	−44,614	−70%	largest sign-flip
skills-quality/sonnet	97,928	109,465	−11,537	**−12%**	sign-flip
skills-quality/opus	56,615	53,895	+2,720	+5%	marginal
skills-quality/haiku	81,544	67,623	+13,921	+17%	save
Aggregate	**629,795**	**613,918**	**+15,877**	**+2.5%**	

3 of 6 cells went negative. The +2.5% headline was honest-but-small, and, as the section above explains, it is a skill-vs-cold number inflated by SKILL.md overhead, not the optimization's effect. The value of this round is the forensic pass: it produced the three cost-trap mechanisms, each visible in the transcripts and each traced to a specific line of procedural prose.

Round 2: post-fix validation on skills-freshness

`claude-skills` PR #18 tightened `skills-freshness/SKILL.md` step 3 with the exact guard wording that targets the three mechanisms: `limit=80`, `no deep read`, `max 3 traces total, don't spelunk`. Round 2 re-ran the skills-freshness A/B (`skills-quality/SKILL.md` was unchanged, so skipping its cells avoided pure measurement noise).

Cell	R1 %	R2 %	Swing	Note
skills-freshness/sonnet	+43%	+14%	−29pp	still positive, BEFORE got cheaper
skills-freshness/opus	−21%	**+1%**	+22pp	sign-flip cleared

Cell	R1 %	R2 %	Swing	Note
skills-freshness/haiku	−70%	+20%	+90pp	largest single swing in study

Aggregate skills-freshness: +2.7% → +11%. Hypothesis confirmed on all three cells. The `limit=80` guard killed haiku's 233K-char full-file read (mechanism 1). The "max 3 traces, don't spelunk" cap stopped opus's source-repo hunt (mechanism 2). Sonnet's BEFORE happened to be cheaper this run (194,751 → 106,046), so the relative save shrank, but sign and direction held. That last point is the methodological weakness: the swing carries the cold arm's run-to-run variance (see Method → Pin the BEFORE arm). The **+90pp** on haiku survives it because it dwarfs the variance; the **−29pp** on sonnet is mostly baseline noise, not a real regression.

Round 3: rigor-exempt label test on content-audit

The new cost-trap rules flagged a group of "rigor-exempt" skills where we dismissed the findings on the assumption that cost-IS-the-point: five tracked in issue #20, drawn from the broader seven-skill set this study's Round-1 scope skipped (`rigor_exempt_skills_skipped` in the raw JSON). Round 3 tested that assumption on `content-audit` (Spacepotatis project skill, single-phase, no nested sub-agents, the cleanest test case in that skipped set).

Cell	BEFORE	AFTER	Saved	%
content-audit/sonnet	138,012	142,301	−4,289	−3%
content-audit/opus	162,332	149,141	+13,191	+8%
content-audit/haiku	93,281	98,137	−4,856	−5%
Aggregate	**393,625**	**389,579**	**+4,046**	**+1%**

Finding: content-audit sat at break-even, not deep-negative as the rigor-exempt label assumed. The label was inflated. Wholesale exemption from cost-trap rules would silence a genuine signal.

Round 4: post-fix re-run of content-audit

Added "at most 5 traces, batch into a single grep with alternation if more, don't spelunk per-id" to step 2 of content-audit's SKILL.md.

Triage thresholds, stated up front (these are decision rules for "keep the fix and move on?", not measurement gates, see Method → On the PASS/MIXED/FAIL thresholds):

- **PASS**: ≥ 2 of 3 cells flip AND aggregate > +5%
- **MIXED**: 1 cell flips AND aggregate moves but stays < +5%
- **FAIL**: 0 cells flip OR aggregate regresses

Cell	R3 %	R4 %	Swing	Flipped?
content-audit/sonnet	−3%	**−9%**	−6pp	no (within noise)

Cell	R3 %	R4 %	Swing	Flipped?
content-audit/opus	+8%	+10%	+2pp	no (already +)
content-audit/haiku	−5%	**+11%**	**+16pp**	**YES (neg → pos)**
Aggregate	+1%	+4%	+3pp	,

Verdict: MIXED. Haiku's +16pp mirrors Round 2's freshness/haiku result, smaller models reliably benefit from bounded procedural language. The honest reading of the other two cells: sonnet's −6pp and opus's +2pp are small **swings**, and a swing carries more noise than a level (see Method → Pin the BEFORE arm), so neither is a trustworthy signed *change*, only haiku's +16pp clears it. Opus was already positive going in, so only sonnet and haiku were even candidates to flip; one did.

Two N=1 caveats apply to this table: R3 and R4 used independent BEFORE arms (content-audit's cold-walk baseline drifted +7–13% per model between the two runs), so the per-cell pp swings fold in baseline variance, not just the fix's effect, the opus +2pp in particular sits inside its own ~13% baseline drift. The robust signal is the haiku sign-flip, which is large enough to survive that noise.

Fix shipped anyway as Spacepotatis PR #280.

Round 5: re-validation of both new skills

The question: are Rounds 1 + 2 actually repeatable, or did we tell a coherent story from N=1 noise?

Triage thresholds, stated up front:

- **PASS**: skills-freshness ≥ 2 cells positive AND skills-quality direction matches Round 1 AND aggregate > 0
- **MIXED**: one of the two passes
- **FAIL**: both deviate from prior rounds

Cell	R1 %	R2 %	R5 %	Notes
skills-freshness/sonnet	+43%	+14%	+1%	positive each round; +1% is inside the noise floor
skills-freshness/opus	−21%	+1%	−1%	sign-flip cleared; ±1% is no signed signal
skills-freshness/haiku	−70%	+20%	+48%	consistent strong saver post-fix
skills-quality/sonnet	−12%	,	**+2%**	flipped positive (inside noise floor; no R2 data)
skills-quality/opus	+5%	,	−62%	**N=1 anomaly**, BEFORE was 31K (short-circuited)
skills-quality/haiku	+17%	,	+54%	strongest haiku save in study
Aggregate (6 cells)	,	,	**+16%**	strongest aggregate of any round

Verdict: MIXED per the strict goal. skills-freshness passed (2 of 3 cells positive). skills-quality didn't directionally match Round 1 (1 of 3 cells match sign). But aggregate is the strongest of any round (+16% across the 6 cells / 12 agents).

The +16% is concentrated, not broad-based. The two haiku cells (skills-freshness +52,289, skills-quality +55,623) sum to ~108K (more than the entire net save of 90,288), while the other four cells net **−17,624** (freshness/sonnet +1,548, freshness/opus −1,921, quality/sonnet +2,195, quality/opus −19,446). Read "strongest aggregate" as "two strong haiku saves outweighing four flat-to-negative cells," not an across-the-board gain, and note that three of those four other cells (everything but quality/opus) are inside the noise floor. This is what ratio-of-sums weighting surfaces (see Method → How aggregates are computed).

The opus/skills-quality −62% is an N=1 anomaly: BEFORE cell was 31K tokens (opus short-circuited cold without thoroughly auditing); AFTER (51K) was the more representative number.

Cross-round synthesis

The durable claim

Bounded, scoped procedural language in a SKILL.md **reliably reduces Haiku-class token use; on Opus-class the effect is inside the noise.** That is the one result this data supports across all five rounds, and it is supported by *swings with identified mechanisms*, not by the aggregate percentages:

- skills-freshness/haiku **−70%** → **+20%** (+90pp, R1→R2), mechanism = unlimited read, fix = `limit=80`.
- content-audit/haiku **−5%** → **+11%** (+16pp, R3→R4), mechanism = uncapped per-id tracing, fix = the trace cap.
- skills-quality/haiku **+17%** → **+54%** (R1→R5), same investigation-collapse pattern.

There is **no portfolio-level save claim** in this data. Three skills, N=1 per cell.

Where the numbers are trustworthy, and where they are not

- **Haiku is the round's largest-saving cell by % saved in three of five rounds** (R2 +20%, R4 +11% flip, R5 +54%). By absolute tokens R4's largest is opus; R2 and R5 hold under both metrics. In the two rounds where haiku isn't the largest (R1, R3) it was a *negative* cell: pre-fix, bounded guidance can cost haiku more than it saves (R1 freshness/haiku −70%). So "reliable beneficiary" describes the **post-fix** pattern.
- **Opus is the most variable model**: single-cell swings up to **67pp** between rounds (skills-quality/opus +5% in R1 → −62% in R5). Most individual Opus cells are inside the noise floor; do not read them as confident signal.
- **Aggregate "growth" is a composition artifact.** The headline aggregates (R1 +2.5%, R2 +11%, R5 +16%) are *not* a like-for-like series: R1 and R5 are 6 cells (freshness + quality), R2 is 3 cells (freshness only, quality was skipped), and Rounds 3–4 measure a different skill (content-audit, +1% / +4%). The comparable freshness-only ratio-of-sums climbs cleanly: **+2.7% (R1) → +10.7% (R2) → +14.9% (R5)**. That composition-matched series is the real trend; the raw progression also reflects changing composition.

Implications for the cost-trap rules

The three rules added in PR #18 (`unlimited_read_in_procedure`, `uncapped_followup`, `batch_invitation`) detect a real class of waste, validated on at least one cell each. But the rule messages slightly overclaim universality. A future iteration could note "fix likely helps haiku-class models substantially; may modestly hurt opus-class models that don't over-execute" so users know what to expect.

The five remaining rigor-exempt skills in issue #20 are still labeled exempt **by assumption only**: each would benefit from a Round-3-style test before the dismissal is treated as durable. Round 3 on content-audit showed the label was inflated for at least one such skill.

Method

A/B design (per cell)

Two fresh sub-agents per cell, same task statement, same target directory, same hour, same model. Differ only in whether the agent received the current SKILL.md.

- **BEFORE** arm: task only, no procedural guidance. Solves cold.
- **AFTER** arm: task plus current SKILL.md. Follows the procedure.

Each agent returned a structured report (skills examined, findings count, tool-use estimate, approach, summary) via JSON schema for consistent comparison.

Token accounting

Per-agent JSONL transcripts under `~/.claude/projects/ … /subagents/workflows/*/agent-*.jsonl`. For each assistant message, summed `usage.input_tokens + usage.cache_creation_input_tokens + usage.output_tokens`, deduped by `(sessionId, requestId)` (per agent, so `requestId` alone is equivalent). `cache_read_input_tokens` excluded. Those were paid upstream. Same convention as `scripts/build-review-stats.mjs`.

How aggregates are computed

Every round's aggregate `%` is **ratio-of-sums**: net tokens saved across all cells divided by total BEFORE tokens, volume-weighted, so high-baseline cells count more. It is **not** the mean of the per-cell percentages, and the two diverge sharply at N=1. Round 1's +2.5% ratio-of-sums becomes **−6.3%** as an unweighted mean of its six cell percentages; Round 5's +16% becomes **+7%**. Ratio-of-sums answers "did this batch of runs get cheaper overall," which is the right question for a portfolio-cost lens, but every headline `%` here should be read as a token-weighted figure, not a typical-cell figure.

The noise floor is estimated, not measured

The ~10% discrepancy noted in the workflow-record footnote below is an **accounting difference between two token-counting methods** (the harness-reported workflow total vs the per-cell JSONL recompute), **not** a measured per-arm noise floor. The true run-to-run noise on a single N=1 arm is **unmeasured**: pinning it down would require repeated identical runs, which this study did not do. So when the text calls a ±1–2% cell "inside the noise floor," that floor is itself an estimate (≈ the BEFORE-arm drift seen between rounds, +7–13% on content-audit), and the real floor could be wider. The practical rule applied throughout: **cells within roughly ±2% carry no signed signal**: direction at best.

Pin the BEFORE arm

This study re-ran the cold (no-SKILL.md) baseline **every round**, in the same hour as that round's AFTER arm. The benefit is that model drift over calendar time is neutralised: BEFORE and AFTER always share the same model build. The cost is that the cold arm's run-to-run variance lands **directly in the swing**, which is the number that carries the thesis: e.g. skills-freshness/sonnet's BEFORE moved 194,751 → 106,046 between R1 and R2, which is most of why its swing read −29pp despite no real regression.

Because BEFORE is *by definition* "no SKILL.md," it does **not** change as the skill evolves. A stronger design would run a small **averaged** cold reference per (model, task) (several cold draws, not one), and pin it across rounds. Then the swing reduces to `(AFTER₁ − AFTER₂) / baseline`, isolating the optimization, and the per-round BEFORE noise drops out. Averaging matters: pinning a single N=1 cold draw would just propagate that one draw's idiosyncrasy into every swing.

The tradeoff is explicit: a pinned baseline reintroduces model-drift risk if the AFTER arms are measured weeks apart. Within a same-hour / same-week window that risk is negligible, which is the regime where pinning is the better choice. This study chose re-run-each-round to be maximally conservative about drift, at the cost of baseline variance in the swing. **Note also that a swing is a difference of two N=1 ratios, so it carries *more* noise than a level, not less**, which is why only the large swings (+90pp, +16pp) are headline-safe and the small ones (sonnet −6pp, opus +2pp) are the worst of both worlds. Pinning the baseline is what would make small swings trustworthy.

On the PASS/MIXED/FAIL thresholds

The Round 4 and Round 5 criteria were stated **before** the results were known. That pre-registration is deliberate and its value survives N=1: it prevents post-hoc goalpost-moving (choosing the verdict that flatters the result). What it does **not** buy is precision. A `> +5%` cutoff at N=1, with a noise floor of unknown width below it, is a **triage rule** ("is this fix good enough to keep and move on?"), not a measurement gate ("the true effect exceeds 5%"). Read the verdicts as decisions under cheap evidence, and read any cell inside ±2% as direction-at-best rather than a signed pass/fail.

Why same-model, same-hour

The skill-registry's aggregated stats fold in transcripts from different sessions, different models, and different weeks. Useful for portfolio-level cadence; dangerous as an isolation of any single optimization's effect. This study pins the model, task, and hour so any delta is attributable to the SKILL.md condition rather than to model drift or cross-session cache state, though, per the two-questions split above, the within-round delta still includes the skill's fixed loading overhead, so it is not *purely* the optimization.

Workflow run record

Round	Workflow ID	Agents	Wall-clock	Subagent tokens
1	`wf_69f7e105-98f`	12	244 s	1,131,817
2	`wf_8323de85-910`	6	190 s	642,172
3	`wf_1f7f067c-791`	6	573 s	758,141
4	`wf_9c93fac3-e6b`	6	610 s	828,450
5	`wf_0d56c8c9-5bc`	12	212 s	1,122,667
Total		**42**	**~30 min**	**~4.48M**

The **Subagent tokens** column is each run's workflow-harness-reported total, captured from the `wf_*` run record at completion. It uses a different accounting/dedupe scope than the per-cell BEFORE/AFTER sums recomputed from the JSONL transcripts in the round tables, so the two do not tie out exactly (they differ by up to ~10% per round, in both directions). The per-cell token figures are the authoritative measurement behind every save and percentage in this study; this column is a rough cost-of-running-the-study tally only.

Fixes shipped between rounds

- `MikkoNumminen/claude-skills` PR #18: cost-trap rules + skills-freshness SKILL.md tightening (between Round 1 and Round 2)
- `MikkoNumminen/claude-skills` PR #19: clears 3 cleared-this-round findings on ai-codegen-smell-audit, readme-drift-sync, skill-calibration (between Round 2 and Round 3)
- `MikkoNumminen/claude-skills` issue #20: rigor-exempt dismissal tracking
- `MikkoNumminen/Spacepotatis` PR #280: content-audit cap (between Round 3 and Round 4)

What this study does NOT claim

- That the within-round aggregate is the optimization's effect. The optimization's effect is the **swing**; the aggregate is **skill-vs-cold**, and it bundles in SKILL.md-loading overhead.
- That the +16% aggregate generalizes to the broader portfolio. Three skills were measured; this study isn't the calibration.
- That the "minimize LLM tokens" principle is invalidated. Three of six original cells confirmed it; the conditions matter.
- That maintenance cost of writing the new cost-trap rules and the SKILL.md edits is folded in. It isn't.
- That per-model dollar pricing is included. Raw token totals only.
- That a second N=1 re-run would land within 5% of these numbers. With per-cell variance up to 67pp on Opus, the precision bound is wider than we can pin from one observation.