
Do Energy-Aware Agent Budgets Survive a Deployment Change? A Measured-Energy Study of Target-Regime Cost Calibration

Han Shaun Lee
Stanford University
hanl@stanford.edu

Abstract

Energy-aware agents decide whether to stop, sample another solution, or call a verifier under a per-question energy budget. A controller tuned in one serving regime is usually deployed in another, and a different GPU or batching concurrency changes the joules per call by up to $4.7\times$. We ask whether recalibrating only the controller’s cost model on a few target-regime questions preserves its accuracy–energy–violation trade-off without retraining the language model. We measure a frozen Qwen2.5-1.5B-Instruct agent (sampled chain-of-thought attempts plus self-verification) in four regimes (A10 at concurrency 8 and 64, H100, L4) with the NVML energy counter over long windows, 16,190 calls per regime, and evaluate frozen protocols by replay on held-out questions. The pre-registered primary test with the original controller is null: it is conservative by construction, because within a regime per-call energy is almost deterministic in output tokens and its cost-residual reserve misses the dominant risk, next-attempt length. With a token-quantile predictive reserve, chosen on source validation data only, calibration helps in both directions. A pre-registered confirmatory study on 500 never-used questions measured in new containers confirms this: calibration on $n=20$ questions cuts budget violations by 1.8–16.3 pp in all six shifts to a more expensive regime (accuracy change -0.9 to 0.0 pp) and raises accuracy by 0.4–2.0 pp in four of six shifts to a cheaper one. Regime energy constants replicate across containers within 2.2%, and calibrating from another container’s traces works as well. A controller without the weak self-verifier uses 18–23% less energy than a budget-aware fixed- k sampler at 0.2–0.6 pp lower accuracy in three of four targets (pre-registered margin 1.5 pp), but misses the margin in the cheapest regime. Conclusions hold under token-share and net-above-idle energy, and a live (non-replay) run of the frozen controllers reproduces the replayed energy within 2% and the calibration effect on violations (-39 and -47 pp).

1 Introduction

Test-time compute (repeated sampling, voting, self-verification) is now a standard way to buy accuracy from a fixed model [4, 8]. When the currency is energy rather than tokens, the exchange rate between an action and its cost depends on where the model is served. The Intelligence-per-Watt study [1] showed large differences in accuracy per joule across local accelerators, and the CS224R project of Wang and Davion [3] made energy a first-class post-training objective. Neither asks what happens to a *budget controller* when the deployment regime changes after the controller is tuned.

This paper studies that narrow question with measured energy. A controller observes only what a deployed agent can know (answers of past attempts, verifier verdicts, past chosen calls’ measured cost, remaining budget) and chooses STOP, CONTINUE (one more sampled solution) or VERIFY (ask the same model to grade the current answer). It uses a learned gain model (expected change in correctness of the voted answer) and a per-regime cost model (joules per call as a linear function of output tokens) with an uncertainty reserve. The candidate method recalibrates only the cost model on n target-regime questions; the gain model and thresholds stay frozen from the source regime.

Contributions. (1) A measured trace set: four serving regimes (A10, H100 and L4 GPUs, and two batching concurrencies on the same A10), identical question seeds, NVML counter energy over 95–735 s windows, two attribution rules, idle power, p95 latency. (2) A replay harness that enforces no-label/no-future-cost observations by construction, with leakage, attribution and cost-accounting tests. (3) A frozen, pre-registered held-out evaluation of source-only versus calibrated controllers against fixed- k self-consistency, token budgets, a static energy budget, and an oracle (diagnostic). (4) A 4×4 source–target transfer matrix showing that the value of target calibration is asymmetric: it removes most avoidable budget violations when the target is more expensive than the source and is useless when it is cheaper. (5) A diagnosis of why the learned controller is conservative, from source-only measurements, and a predictive token-quantile reserve that fixes it, evaluated with a fair budget-aware baseline. (6) A pre-registered confirmatory study on fresh questions measured in new containers, energy-definition sensitivity analyses, and a live (non-replay) run that validates the replay methodology.

2 Related work

Energy measurement of LLM inference. Intelligence per Watt [1] measures accuracy and energy of local inference systems with an NVML/RAPL energy monitor (50 ms default sampling in the audited snapshot 3f89981); OpenJarvis [2] builds a local-agent stack that treats energy as an evaluation constraint. Samsi et al. [9] benchmark LLM inference energy on datacenter GPUs; Zeus [10] and DynamoLLM [11] optimise GPU energy through power limits, frequency and configuration. We change no power limits and measure at the board level only. Wang and Davion [3] report that output tokens explain most energy variance on H100 and use an *assumed* BF16→FP8 correction factor in one comparison; we use one precision (BF16) for all controlled comparisons and apply no correction factors. **Adaptive test-time compute.** Self-consistency [4], Adaptive-Consistency [5] and early-stopping self-consistency [6] stop sampling once votes agree; FrugalGPT [7] cascades models under a dollar budget; compute-optimal test-time scaling [8] allocates samples by estimated difficulty. These works fix the cost of an action; our question is what happens when that cost moves with the deployment regime. **Serving.** All traces use vLLM [12]; batching concurrency is its `-max-num-seqs`.

3 Method

Agent and actions. For each question the agent must take one attempt, then repeatedly chooses STOP, CONTINUE (attempt $k+1$, max 5) or VERIFY (grade the currently selected answer; one verification per attempt). The answer is the plurality of parsed attempt answers, with a verifier verdict adding (+1) or removing (−1) one vote. The episode’s energy is the sum of measured energies of the calls actually chosen, including verification calls. **Observation.** $o_t = (k, \text{parsed answers, completion-token counts and truncation flags of past attempts, verdicts, spent joules, spent latency, remaining budget})$. Correctness labels, future calls, and costs of unchosen actions are not observable; the replay environment holds the trace and reveals a call’s cost only after it is chosen (`src/p01/controller.py`; tests in `tests/test_p01.py`). **Gain model.** Two gradient-boosted regressors (one per action) predict the change in correctness of the selected answer from 14 regime-agnostic state features (agreement structure, verification state, length structure), trained on random-action rollouts over 500 source-training questions (24 rollouts per question), three seeds. **Cost model.** $\hat{E}_a(\text{tokens}) = \alpha_a \cdot \text{tokens} + \beta_a$ for $a \in \{\text{attempt, verify}\}$, fit by least squares; σ_a is the residual s.d. The next attempt’s tokens are predicted as the past mean shrunk to the source mean. **Controller.** Among actions with $\hat{E}_a + z\sigma_a \leq \text{remaining budget}$ and predicted gain $g_a > \lambda$, choose $\arg \max g_a / \hat{E}_a$; otherwise stop. *Source-only* uses the source cost model in every regime; *calibrated-*

Table 1: Measured regimes (all 16,190 calls, time-share attribution). Window CV: coefficient of variation of J/token across the five attempt windows (repeat-measurement noise).

regime	idle W	mean W	tok/s	attempt J	J/tok	p95 lat. (s)	window CV
S (A10, c=8)	56.6	150	865	41.6	0.173	4.10	0.2%
T1 (A10, c=64)	60.1	149	4057	8.8	0.037	6.74	0.7%
T2 (H100, c=8)	132.4	309	2672	27.9	0.116	1.31	0.7%
T3 (L4, c=8)	33.5	72	542	32.1	0.133	6.57	0.1%

n refits (α, β, σ) on all calls of n questions from the target regime’s calibration split. (z, λ) are selected on the source validation split only.

4 Experimental setup

Model, data, splits. Qwen2.5-1.5B-Instruct (HF revision 989aa79, Apache-2.0), BF16, vLLM 0.21.0, temperature 0.7, top- p 0.95, max 512 tokens per attempt and 200 per verification, fixed per-(question, round) seeds identical across regimes. GSM8K test [13] (1,319 questions) is split at the question level (seed 20261021) into source-train 500, source-validation 200, target-calibration 200, target-test 419; every call of a question, in every regime, belongs to one split. SVAMP [14] (300 questions) is a held-out task family never used for fitting or selection. **Regimes.** S: A10 at concurrency 8 (source); T1: A10 at concurrency 64 (same device, batching change; we call this a deployment-regime change, not cross-hardware transfer); T2: H100 80GB at concurrency 8; T3: L4 at concurrency 8. Each regime ran in a fresh Modal container with an isolated GPU (no foreign compute processes before or after), at the provider’s default power limit (A10 150 W, H100 700 W, L4 72 W). **Energy boundary and attribution.** GPU board energy from `nvmlDeviceGetTotalEnergyConsumption` read at window start and end; one window is one round of all 1,619 questions (attempt windows 95–735 s, verification windows 12–55 s). Host CPU, DRAM, and network energy are *not* included. Board power is sampled at 20 Hz (NVML updates at ~ 10 Hz) and each sample interval’s energy is split equally among requests in flight (*time share*); per-call values are rescaled so a window’s calls sum exactly to its counter reading. Token-share attribution is recorded as a sensitivity. A telemetry pilot found integrated sampled power within 0.7–7.7% of the counter over an 8 s load; idle power was measured for 15 s with the model loaded. We report total GPU energy (not net of idle). **Protocol.** Budgets $B \in \{1.5, 2, 3, 4, 6, 8\} \times$ the median source attempt energy (42.5 J), i.e. 64–340 J, absolute in every regime. (z, λ) were chosen from $\{0, 0.5, 1, 2\} \times \{0, 0.005, 0.01, 0.02\}$ to maximise source validation accuracy subject to violation rate $\leq 5\%$ pooled over budgets, giving $z=0, \lambda=0.01$; the protocol was written to `protocol.json` before any target question was evaluated. Calibration draws: $n \in \{5, 10, 20, 40, 80\}$, five draws, primary $n=20$. The summary rules (unit, averaging over seeds and draws, decision rule for the headline) were written to `ANALYSIS_PLAN.md` before aggregation. **Baselines.** Fixed $k \in \{1..5\}$ self-consistency; fixed token budgets (300–1800 tokens); a static energy budget (continue while spent $< B$); source-only controller; calibrated controller without reserve and without verify; a cost model fit on 500 target questions (reference); and an oracle that sees labels and realised costs (diagnostic only, unattainable). **Statistics.** Unit = question. A question’s metric is averaged over calibration draws within a gain seed, then over the three seeds; 95% paired bootstrap CIs over questions (5,000 resamples). **Cost.** All Modal GPU jobs for this study (pilots, smoke, four trace collections) are logged in `cost_ledger.csv` at 10.66 USD (conservative: client wall time); controller replay is CPU-only.

5 Results

The regime shift is large and well above measurement noise. Table 1: J per output token ranges from 0.037 (A10, concurrency 64) to 0.173 (A10, concurrency 8), a $4.7\times$ change on the *same* device. Repeat windows of the same regime agree to within 0.1–0.7% CV, so regime differences are 2–3 orders of magnitude above window-to-window noise. Accuracy of single attempts is stable across regimes (0.639–0.645) even though batched numerics change 25–27% of sampled answers. Both A10 regimes run at the 150 W power limit.

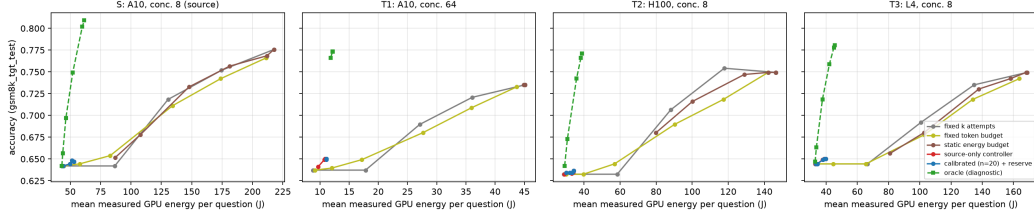


Figure 1: Accuracy vs. mean measured GPU energy per question on the 419 held-out GSM8K questions, one point per budget. The learned controllers (red, blue) sit at the low-energy end with ≈ 1.2 attempts on average; calibration does not move them. The oracle (green, diagnostic) shows large headroom at near-single-attempt energy.

Table 2: Primary pre-registered comparison: calibrated ($n=20$) minus source-only, pooled over the six budgets, paired 95% bootstrap CIs over questions. S is the in-regime reference.

target	test set	Δ accuracy (pp)	Δ energy (J)	Δ violation (pp)
S (A10, c=8)	GSM8K	+0.00 [+0.00, +0.00]	+0.00 [-0.00, +0.00]	+0.00 [+0.00, +0.00]
T1 (A10, c=64)	GSM8K	+0.15 [-0.12, +0.44]	+0.41 [+0.25, +0.59]	+0.00 [+0.00, +0.00]
T2 (H100, c=8)	GSM8K	+0.05 [-0.03, +0.15]	+0.60 [+0.45, +0.77]	+0.52 [+0.30, +0.77]
T3 (L4, c=8)	GSM8K	+0.00 [-0.04, +0.04]	+0.18 [+0.11, +0.25]	+0.17 [+0.08, +0.28]
S (A10, c=8)	SVAMP	+0.00 [+0.00, +0.00]	+0.00 [+0.00, +0.00]	+0.00 [+0.00, +0.00]
T1 (A10, c=64)	SVAMP	+0.09 [-0.17, +0.39]	+0.28 [+0.07, +0.61]	+0.06 [+0.00, +0.15]
T2 (H100, c=8)	SVAMP	+0.00 [-0.06, +0.06]	+0.18 [+0.09, +0.28]	+0.19 [+0.06, +0.35]
T3 (L4, c=8)	SVAMP	+0.04 [+0.00, +0.11]	+0.09 [+0.04, +0.14]	+0.07 [+0.02, +0.15]

Primary result: target calibration has no measurable benefit. Table 2: on GSM8K the accuracy difference is ≤ 0.15 pp with every CI containing 0; energy rises by 0.2–0.6 J; violation rates rise slightly in T2 and T3 (CIs exclude 0, magnitudes < 0.6 pp). The same holds on SVAMP. By the decision rule fixed in the analysis plan, the headline is null. The calibration-size curve (Appendix, Fig. 8) is flat in n .

The frozen controller is conservative, and calibration cannot change that. Table 3 and Fig. 1: both learned controllers take ≈ 1.2 attempts and reach 0.63–0.65 accuracy, close to fixed $k=1$, whereas fixed $k=5$ reaches 0.735–0.776 at $4\text{--}5\times$ the energy. In T1 every budget is slack (fixed $k=5$ costs 45 J against $B \geq 64$ J), yet neither controller spends it, because the stop decision is $g_a \leq \lambda$ and λ is not denominated in joules: a cheaper regime makes more actions *feasible* but does not make any action *worth* taking. Budget-agnostic baselines violate the budget heavily wherever it binds (fixed $k=5$: 63–94% in S, T2, T3); the static energy budget violates by construction because it spends until the budget is exhausted.

Why the reserve did nothing. With time-share attribution, per-call energy within a regime is nearly a linear function of output tokens (GSM8K residual s.d. 0.4–1.3 J; $\sigma_{\text{attempt}} = 0.60$ J on source-train). The reserve $z\sigma$ is therefore < 1 J, while the uncertainty that actually causes overruns is the next attempt’s length (s.d. ≈ 99 tokens, ≈ 17 J in S). This is visible *before* test: the source-validation selection table is flat in z , and the violation constraint was met only by raising λ , which bought safety by suppressing extra attempts. Removing the reserve changes nothing on test (Appendix Table 8).

Direction of the shift. All three targets of the primary setting are cheaper per call than the source (8.8, 27.9 and 32.1 J vs. 41.6 J per attempt), so the source cost model over-predicts cost in every target and the source-only controller errs on the safe side; its violation rate in targets (0.0–0.6%) is lower than in the source (3.3% pooled).

Transfer matrix: calibration matters when the target is more expensive. To test the configuration the primary setting cannot probe, each regime served once as the source, with the v1 recipe re-run unchanged (models on that source’s train split, (z, λ) on its validation split, budgets $\{1.5..8\} \times$ its median attempt energy, protocol frozen before any target episode; this extension was logged before it ran). Fig. 2, Table 4: in all six cells where the target is more expensive than the source, calibration reduces violations (T1→S -7.0 pp, T1→T2 -7.7 , T1→T3 -6.6 , T2→S -6.9 , T2→T3 -2.0 ,

Table 3: All policies at $B = 127.4\text{J}$ ($3\times$ median source attempt), held-out GSM8K: accuracy, mean J, violation %.

policy	S (A10, c=8)			T1 (A10, c=64)			T2 (H100, c=8)			T3 (L4, c=8)		
	acc.	J	viol.	acc.	J	viol.	acc.	J	viol.	acc.	J	viol.
fixed $k=1$	0.642	43.1	0.0	0.637	8.8	0.0	0.632	28.8	0.0	0.644	32.8	0.0
fixed $k=3$	0.718	130.5	50.4	0.690	27.1	0.0	0.706	88.1	9.8	0.692	100.8	18.4
fixed $k=5$	0.776	217.6	93.8	0.735	45.1	0.0	0.749	146.5	63.5	0.749	168.7	77.1
token budget 600	0.654	82.9	0.7	0.649	17.2	0.0	0.644	56.9	0.0	0.644	65.5	0.0
static energy budget	0.733	147.7	93.8	0.735	45.1	0.0	0.747	128.9	63.5	0.730	137.8	77.1
source-only controller	0.644	49.8	3.0	0.650	10.9	0.0	0.633	33.2	0.0	0.649	37.7	0.2
calibrated ($n=20$)	0.644	49.8	3.0	0.650	11.1	0.0	0.634	33.5	0.5	0.649	38.0	0.4
calibrated, no reserve	0.644	49.8	3.0	0.650	11.1	0.0	0.634	33.5	0.5	0.649	38.0	0.4
calibrated, no verify	0.644	49.9	3.1	0.650	11.6	0.0	0.649	34.7	1.3	0.648	38.7	1.1
full-target cost model	0.644	49.8	3.0	0.650	11.1	0.0	0.634	33.5	0.5	0.649	38.0	0.4
oracle (diagnostic)	0.697	46.4	0.0	0.773	12.2	0.0	0.742	35.8	0.0	0.718	37.6	0.0

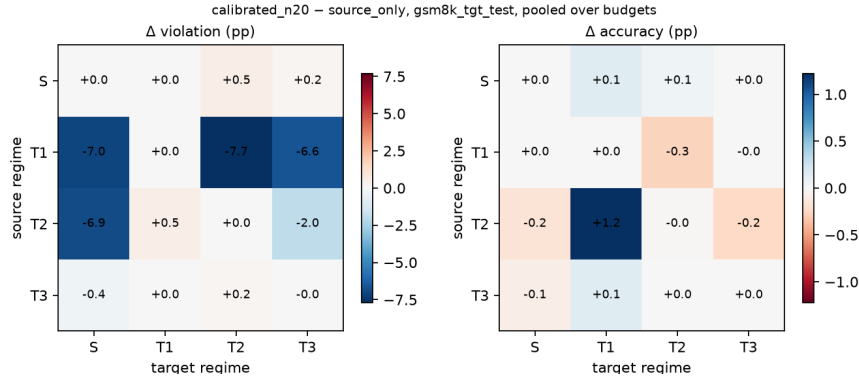


Figure 2: Transfer matrix (held-out GSM8K, pooled over each source’s budget grid): calibrated ($n=20$) minus source-only. Rows: source regime (protocol re-selected on that source’s validation split); columns: target. Regimes ordered by cost per attempt: T1 (8.8J) < T2 (27.9) < T3 (32.1) < S (41.6). Cells above the anti-cost order (source cheaper than target) show violation reductions.

T3→S -0.4 ; all CIs exclude 0) with accuracy changes between -0.27 and 0.00 pp and *lower* energy (-0.2 to -4.3 J). Much of the remaining violation is unavoidable: budgets set in a cheap source are below the cost of a single attempt in the expensive target for 4–62% of questions (the fixed- $k=1$ floor). Measured against that floor (a post-hoc decomposition), calibration removes 35–96% of the *avoidable* violations (source-only exceeds the floor by 1.0–9.7 pp, calibrated by 0.3–3.0 pp). In the six cheaper-target cells calibration never reduces violations ($+0.0$ to $+0.5$ pp) and changes accuracy by ≤ 0.15 pp except T2→T1 ($+1.2$ pp [$+0.6, +1.9$] at $+2.1$ J). SVAMP shows the same sign pattern (Appendix Table 9). With only one source–target pair per cell and four regimes, the asymmetry is a measured pattern over 12 cells, not a law.

Round 2: a predictive reserve. The v1 reserve $z\sigma$ covers only the cost-model residual. The revised controller treats an action as feasible iff $\alpha(\hat{t} + q_\tau(k)) + \beta + \sigma \leq$ remaining budget, where $q_\tau(k)$ is the τ -quantile of next-attempt token residuals given k attempts, estimated on the source’s training split ($q_{0.8} \approx 50$ –65 tokens, $q_{0.99} \approx 224$ –232 tokens on S); (τ, λ) was selected on each source’s validation split with the v1 rule, giving $\tau=0.8$, $\lambda=0$ for every source (the predictive reserve meets the 5% violation target without suppressing attempts). We add a fair simple baseline, *budgeted-k*: keep sampling while the same τ -reserve fits (no gain model, no verifier; τ selected on validation, 0.8–0.9). The plan and arms were logged before running. Fig. 3: calibration now helps in *both* directions. In the six cheaper→more-expensive cells it cuts violations by 14.7 (T1→S), 17.6 (T1→T2), 16.7 (T1→T3), 5.4 (T2→S), 1.4 (T2→T3) and 4.0 pp (T3→S), all CIs excluding 0, at accuracy changes of -0.4 to 0.0 pp. In the six more-expensive→cheaper cells it raises accuracy by $+1.9$ [$+1.1, +2.9$] (S→T1), $+0.4$ [$+0.1, +0.7$] (S→T2), $+0.4$ [$+0.1, +0.8$] (S→T3), $+2.0$ (T2→T1), $+1.0$ (T3→T1) and $+0.2$ [$-0.0, +0.3$] pp (T3→T2), spending 1.4–6.1 J more, with violations rising by 0.1–1.8 pp but staying $\leq 2.8\%$. Calibrating with $n=5$ or $n=80$ questions gives the same results within

Table 4: Transfer matrix, held-out GSM8K, pooled over budgets: violation rate of source-only (b) and calibrated (a), paired differences with 95% CIs, and the violation floor (fixed $k=1$: the mandatory first attempt alone exceeds B).

source→target	b viol. (%)	a viol. (%)	Δ viol. (pp)	Δ acc. (pp)	ΔJ	first-attempt infeasible (%)
S→S (in-regime)	3.3	3.3	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	2.6
S→T1	0.0	0.0	+0.0 [+0.0, +0.0]	+0.1 [-0.1, +0.4]	+0.4	0.0
S→T2	0.0	0.5	+0.5 [+0.3, +0.8]	+0.1 [-0.0, +0.1]	+0.6	0.0
S→T3	0.6	0.7	+0.2 [+0.1, +0.3]	+0.0 [-0.0, +0.0]	+0.2	0.4
T1→S	69.5	62.5	-7.0 [-8.0, -6.2]	+0.0 [-0.1, +0.1]	-3.3	62.2
T1→T1 (in-regime)	4.5	4.5	+0.0 [-0.0, +0.1]	+0.0 [+0.0, +0.0]	+0.0	2.3
T1→T2	52.0	44.3	-7.7 [-8.5, -6.9]	-0.3 [-0.5, -0.1]	-2.9	43.7
T1→T3	57.6	50.9	-6.6 [-7.4, -5.9]	-0.0 [-0.1, +0.1]	-2.5	50.0
T2→S	21.1	14.2	-6.9 [-7.8, -6.1]	-0.2 [-0.4, +0.1]	-4.3	11.4
T2→T1	0.0	0.5	+0.5 [+0.3, +0.8]	+1.2 [+0.6, +1.9]	+2.1	0.0
T2→T2 (in-regime)	5.7	5.7	+0.0 [-0.0, +0.0]	-0.0 [-0.0, +0.0]	-0.0	2.2
T2→T3	8.8	6.8	-2.0 [-2.5, -1.5]	-0.2 [-0.5, -0.0]	-1.2	3.8
T3→S	8.4	8.1	-0.4 [-0.5, -0.2]	-0.1 [-0.2, -0.0]	-0.2	7.4
T3→T1	0.0	0.0	+0.0 [+0.0, +0.1]	+0.1 [+0.0, +0.3]	+0.2	0.0
T3→T2	1.6	1.7	+0.2 [+0.1, +0.3]	+0.0 [+0.0, +0.0]	+0.1	1.1
T3→T3 (in-regime)	2.8	2.8	-0.0 [-0.0, +0.0]	+0.0 [+0.0, +0.0]	-0.0	2.3

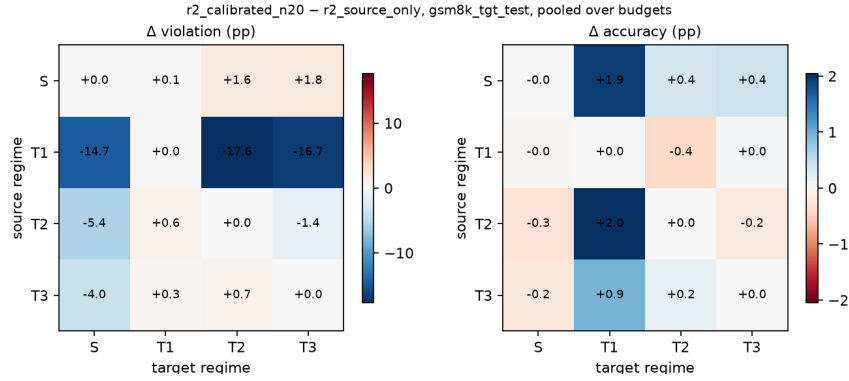


Figure 3: Round 2 transfer matrix with the predictive-reserve controller (held-out GSM8K, pooled over budgets): calibrated ($n=20$) minus source-only.

0.3 pp, and $n=20$ matches the 500-question target cost model within 0.04 pp accuracy (Appendix): because per-call energy is so close to linear in tokens, very little target data identifies the cost model. Budgeted- k shows the same pattern under calibration (violations -2.2 to -22.1 pp toward expensive targets, accuracy $+0.3$ to $+2.4$ pp toward cheap ones; Appendix Table 13), so the value of calibration does not depend on the learned gain model. SVAMP shows the same signs.

Does the learned gain model help? Not as specified. At the same budgets, the calibrated predictive-reserve controller spends 9–25 J less than calibrated budgeted- k but loses 0.6–5.3 pp accuracy (Fig. 4: both lie on the fixed- k frontier). The loss comes from the verify action: the same controller with VERIFY disabled matches budgeted- k accuracy within $[-1.5, +0.5]$ pp in all four targets from source S while spending 13–22 J (16–29%) less (Appendix Table 14). This arm was a pre-declared ablation, but singling it out as the better controller is a post-hoc choice on the test split; we therefore treat it as a hypothesis for a confirmatory run, not as a result.

Verification. The self-verifier says CORRECT on 96% of attempts (INCORRECT precision 0.90, recall 0.11 on S). Disabling VERIFY changes accuracy by -0.03 to $+0.98$ pp (the largest change, in T2, has a CI excluding 0) and energy by $+0.4$ to $+1.4$ J (Appendix): the weak verifier is not worth its cost here.

Latency and overhead. p95 end-to-end latency per question of the learned controllers ranges 1.0–11.8 s across regimes and budgets (Fig. 9); controller decision overhead is 0.01–0.3 ms (CPU, replay), negligible against 0.06–3.7 s per call; we do not add controller CPU energy to the GPU boundary.

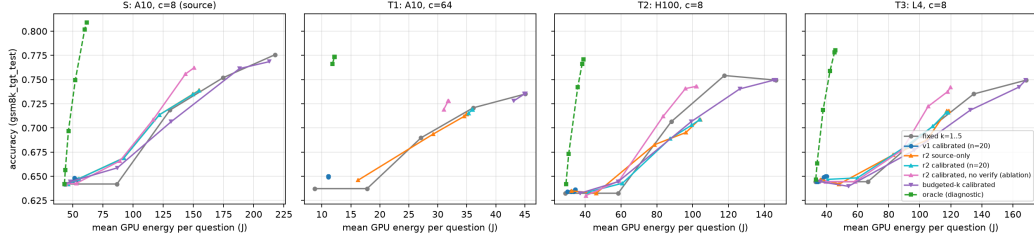


Figure 4: Accuracy–energy frontiers on held-out GSM8K, source S, one point per budget. The predictive-reserve controller (cyan) and budgeted- k (purple) trace the fixed- k frontier; the controller without the verify action (pink; an ablation, identified post hoc) lies above it in S, T2 and T3.

Table 5: Run-to-run replication of regime constants: original run vs. fresh run in a new container (attempt windows).

regime	J/token		Δ	tok/s		idle W	
	original	fresh		original	fresh	original	fresh
S	0.1728	0.1726	-0.1%	865	866	56.6	57.9
T1	0.0367	0.0365	-0.8%	4057	4079	60.1	58.1
T2	0.1160	0.1134	-2.2%	2672	2652	132.4	127.1
T3	0.1325	0.1323	-0.2%	542	543	33.5	29.5

6 Confirmatory study on fresh questions and new containers

Rounds 1–2 reused the same 419 test questions, and the no-verify result was a post-hoc pick. We therefore pre-registered three hypotheses (H1a, H1b, H2; reports/reviews/round_3.md) and collected new measurements: 600 GSM8K *training* questions never used before (seed 20261022; 100 for calibration, 500 for test), measured in fresh Modal containers in all four regimes (6,000 calls each, 0 errors), submitted in random order so that window-tail requests are random questions. Models, (τ, λ) and budgets were re-derived from the original source splits and asserted to be identical to round 2. The GSM8K training split may be in the model’s training data; absolute accuracies (0.70–0.84) are therefore descriptive, while all comparisons are paired on the same questions.

Measurement replicates. Table 5: J/token in a new container differs from the original by -0.1 to -2.2% and throughput by $< 1\%$; regime differences (up to $4.7\times$) are two orders of magnitude larger than instance-to-instance variation.

H1a (supported, 6/6). In every shift to a more expensive regime calibration reduces violations: T1→S -16.0 [$-17.1, -14.9$], T1→T2 -16.1 , T1→T3 -16.3 , T2→S -6.6 , T2→T3 -1.8 [$-2.2, -1.3$], T3→S -3.6 pp, at accuracy changes of -0.9 to 0.0 pp and 4–10J less energy (Fig. 5, Table 10). **H1b (supported, 4/6).** In shifts to a cheaper regime calibration raises accuracy in S→T1 ($+2.0$ [$+1.2, +2.8$] pp), S→T2 ($+0.4$ [$+0.2, +0.7$]), T2→T1 ($+1.9$) and T3→T1 ($+1.3$); S→T3 ($+0.0$) and T3→T2 (-0.1) show no change. Gains are largest when the target is much cheaper (T1). The price is up to 3.9 J more energy and 0.1–0.8 pp more violations (all $\leq 1.5\%$ absolute). **Instance transfer.** Calibrating on the *original* container’s calibration traces of the same regime type instead of the fresh container’s gives the same pattern (Appendix Table 11; differences from fresh-instance calibration ≤ 0.65 pp in accuracy and violations).

H2 (supported, 3/4, with a caveat). Source S, calibrated, verify disabled vs. calibrated budgeted- k : energy -18.2% (accuracy -0.2 [$-0.6, +0.1$] pp) in S, -22.8% (-0.6 [$-1.2, -0.2$]) in T2, -20.9% (-0.6 [$-1.1, -0.2$]) in T3, all within the pre-registered 1.5 pp margin; in T1 the energy saving is -31.7% but accuracy drops 1.4 [$-2.2, -0.7$] pp, outside the margin (Fig. 6, Appendix Table 12). The learned stop rule thus buys energy at a small, measurable accuracy cost; it does not dominate the simple sampler.

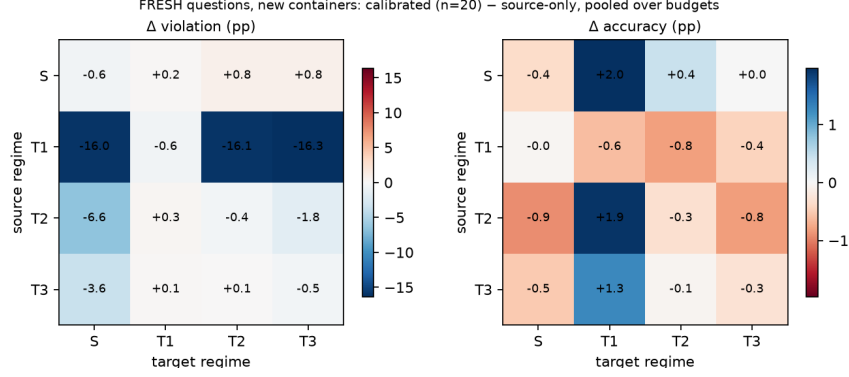


Figure 5: Confirmatory transfer matrix on 500 fresh questions in new containers: calibrated ($n=20$, fresh calibration questions) minus source-only, pooled over budgets. The diagonal now compares a fresh instance against the original-instance cost model.

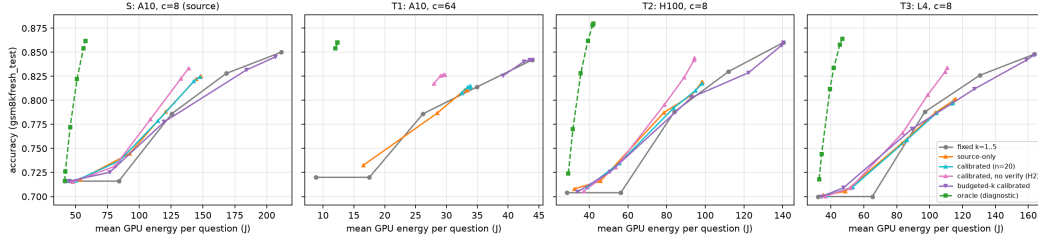


Figure 6: Fresh questions, source S: accuracy–energy frontiers per target regime.

7 Sensitivity to the energy definition

Per-call energies are attributed, and the GPU boundary includes the idle floor (38–47% of window power). We re-ran the complete round-2 protocol (all four sources, re-selection on source validation) under two alternative definitions: token share (window counter energy split in proportion to completion tokens) and net-above-idle (time share scaled by $1 - P_{\text{idle}}/\bar{P}_{\text{window}}$ per window, an approximation that removes the idle floor proportionally; net attempt energies are 25.9, 5.3, 16.0, 17.1 J for S, T1, T2, T3). Selected thresholds were unchanged except $\tau=0.9$ (not 0.8) for source S under token share. Table 6: every qualitative conclusion holds under every definition. Under net-above-idle the T2→T3 shift nearly vanishes (net 16.0 vs 17.1 J), and so does its violation reduction (−0.2 pp), as expected if the benefit tracks the size of the cost shift. Excluding the 32 questions with any call in the last 2% of any measurement window (all SVAMP; no GSM8K test question is affected) changes the SVAMP calibration effects by ≤ 0.4 pp.

8 Live execution: does replay predict deployment?

All results above replay measured calls. A deployed controller changes batch composition and must estimate energy online. We therefore ran the frozen controllers *live* (pre-registered fidelity criteria F1–F3 in EXPERIMENTS.md): source T1 recipe ($\tau=0.8$, $\lambda=0$), live target S (A10, concurrency 8; $4.7\times$ more expensive per token), 300 fresh test questions, budgets 52.9 and 70.5 J, source-only vs. calibrated ($n=20$ fresh calibration questions). Each episode issues its own calls to a live vLLM server with the same prompts and seeds as the traces; the controller sees online time-share energy computed from NVML samples up to each call’s completion (on average 12% below, median 3% below, the counter-rescaled value used for outcomes). Each (arm, budget) is one measurement window (89–144 s). Table 7: replay predicts live behaviour closely. Live mean energy is within −0.3 to +1.6% of replay for every arm and budget (F2), live accuracy within 3.0 pp (F3); the largest gap, source-only at 52.9 J, lies inside the replay CI, and the calibration effect reproduces live: violations −39.0 [−45.3, −32.3] pp (replay −36.3) at 52.9 J and −46.7 [−52.3, −40.3] pp (replay −46.7) at

Table 6: Energy-definition sensitivity (held-out GSM8K, pooled over budgets; round-2 protocol re-run per definition). Fresh-data row from Section 6 for reference.

energy definition	expensive shifts: violations reduced (CI<0)	cheap shifts: accuracy raised (CI>0)	no-verify vs budgeted- k (src S): ΔJ %, Δacc pp
time share (primary)	6/6 (-1.4 to -17.6 pp)	5/6 (+0.2 to +2.0 pp)	-16 to -29%, -0.7 to -0.1
token share	6/6 (-1.8 to -17.2 pp)	5/6 (+0.3 to +2.4 pp)	-19 to -29%, -0.7 to -0.3
net above idle	6/6 (-0.2 to -17.8 pp)	5/6 (+0.1 to +1.9 pp)	-16 to -29%, -0.7 to -0.1
fresh data, time share	6/6 (-1.8 to -16.3 pp)	4/6 (-0.1 to +2.0 pp)	-18 to -32%, -1.4 to -0.2

Table 7: Live vs. replay on the same 300 fresh questions (source T1 controllers, live target S). Replay uses the fresh S traces.

B (J)	arm	mode	accuracy	mean J	violation (%)
52.9	source-only	replay	0.710	58.6	58.3
52.9	source-only	live	0.680	59.5	60.3
52.9	calibrated	replay	0.713	43.6	22.0
52.9	calibrated	live	0.710	44.3	21.3
52.9	cal.-src.	replay	+0.3 pp	-15.1	-36.3 [-41.7, -31.0]
52.9	cal.-src.	live	+3.0 pp	-15.3	-39.0 [-45.3, -32.3]
70.5	source-only	replay	0.713	70.9	53.3
70.5	source-only	live	0.713	71.4	55.0
70.5	calibrated	replay	0.713	45.6	6.7
70.5	calibrated	live	0.713	45.4	8.3
70.5	cal.-src.	replay	+0.0 pp	-25.3	-46.7 [-52.0, -41.0]
70.5	cal.-src.	live	+0.0 pp	-26.0	-46.7 [-52.3, -40.3]

70.5 J, with 15–26 J less energy per question (F1). The number of attempts per episode agrees with replay for 79–100% of questions. All three fidelity criteria are met.

9 Ablations

Appendix Table 8 reports reserve on/off, verify on/off, $n=20$ versus a cost model fit on 500 target questions, and the calibration-size extremes. A cost model fit on $25\times$ more target data behaves identically to $n=20$, confirming that the null is not caused by calibration-set size.

10 Limitations

Boundary. GPU board energy only; no host, DRAM, network, cooling or provider overhead. **Attribution.** Window energy is measured; per-call energy is attributed. Time share is a modelling assumption; during the drain at the end of each window the last $\sim 2\%$ of requests (SVAMP questions, because questions were submitted in file order) receive inflated time shares (residual s.d. 17–37 J), so SVAMP energies carry a measurement-design artefact for ~ 30 of 300 questions; GSM8K test questions are not at window tails. **Replay.** Episodes are replayed from measured calls; one live run (one regime pair, two budgets, 300 questions) agrees with replay, but other regimes were not run live. **Coverage.** One model, one precision, one serving engine, four regimes; in the primary setting all targets are cheaper than the source (the transfer matrix covers both directions). **Infeasible budgets.** When the target is much more expensive, a source-set budget is below one attempt’s cost for many questions; a controller that must answer cannot avoid these overruns, and handling them (routing, refusal, re-budgeting) is out of scope. **Controller design.** The v1 controller’s reserve is the reason for its conservatism; the gain model is myopic (a second attempt cannot change a plurality vote, so its one-step gain is near zero) and the self-verifier is weak (INCORRECT recall 0.11). **Test reuse.** Rounds 1–2 reuse the same 419 test questions under new protocols (each frozen on source splits before its run); the confirmatory study (Section 6) addresses this with fresh questions and new measurements. **Contamination.** The fresh questions come from GSM8K’s training split, which Qwen2.5 may have seen.

11 Reproducibility statement

Code, raw traces (results/p01/*_traces_*/traces/), the frozen protocol, raw episodes (split gzip with SHA-256), analysis plan, and the scripts that generate every table and figure from raw files (src/p01/summarize.py, scripts/make_figures.py, scripts/make_tables.py) are in the repository; commands are in README.md. Upstream pins, model and dataset revisions with SHA-256 and licences are in sources.lock.json.

References

- [1] J. Saad-Falcon, A. Narayan, H. O. Akengin, J. W. Griffin, H. Shandilya, A. G. Lafuente, M. Goel, R. Joseph, S. Natarajan, E. K. Guha, S. Zhu, B. Athiwaratkun, J. Hennessy, A. Mirhoseini, C. Ré. Intelligence per Watt: Measuring Intelligence Efficiency of Local AI. arXiv:2511.07885, 2025. Code: <https://github.com/HazyResearch/intelligence-per-watt> (audited commit 3f89981).
- [2] OpenJarvis authors. OpenJarvis: Personal AI, On Personal Devices. arXiv:2605.17172, 2026. <https://github.com/open-jarvis/OpenJarvis>.
- [3] C. Wang, R. Davion. Intelligence per Joule as a First-Class Post-Training Objective. Stanford CS224R project report, 2026.
- [4] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, D. Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. ICLR, 2023.
- [5] P. Aggarwal, A. Madaan, Y. Yang, Mausam. Let’s Sample Step by Step: Adaptive-Consistency for Efficient Reasoning and Coding with LLMs. EMNLP, 2023.
- [6] Y. Li, P. Yuan, S. Feng, B. Pan, X. Wang, B. Sun, H. Wang, K. Li. Escape Sky-high Cost: Early-stopping Self-Consistency for Multi-step Reasoning. ICLR, 2024.
- [7] L. Chen, M. Zaharia, J. Zou. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. arXiv:2305.05176, 2023.
- [8] C. Snell, J. Lee, K. Xu, A. Kumar. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters. arXiv:2408.03314, 2024.
- [9] S. Samsi, D. Zhao, J. McDonald, B. Li, A. Michaleas, M. Jones, W. Bergeron, J. Kepner, D. Tiwari, V. Gadepally. From Words to Watts: Benchmarking the Energy Costs of Large Language Model Inference. IEEE HPEC, 2023.
- [10] J. You, J.-W. Chung, M. Chowdhury. Zeus: Understanding and Optimizing GPU Energy Consumption of DNN Training. NSDI, 2023.
- [11] J. Stojkovic, C. Zhang, Í. Goiri, J. Torrellas, E. Choukse. DynamoLLM: Designing LLM Inference Clusters for Performance and Energy Efficiency. HPCA, 2025.
- [12] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. Gonzalez, H. Zhang, I. Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. SOSP, 2023.
- [13] K. Cobbe, V. Kosaraju, M. Bavarian, et al. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168, 2021.
- [14] A. Patel, S. Bhattamishra, N. Goyal. Are NLP Models really able to Solve Simple Math Word Problems? NAACL, 2021.
- [15] Qwen Team. Qwen2.5 Technical Report. arXiv:2412.15115, 2024.

A Additional results

Table 8: Ablations, held-out GSM8K, pooled over budgets, paired 95% CIs.

comparison	target	Δ acc. (pp)	Δ energy (J)	Δ viol. (pp)
reserve (z) on vs off	T1	+0.00 [+0.00, +0.00]	+0.06 [+0.01, +0.12]	+0.00 [+0.00, +0.00]
reserve (z) on vs off	T2	+0.00 [+0.00, +0.00]	-0.00 [-0.01, +0.00]	-0.01 [-0.02, +0.00]
reserve (z) on vs off	T3	+0.00 [+0.00, +0.00]	-0.00 [-0.00, +0.00]	-0.00 [-0.01, +0.00]
verify action on vs off	T1	+0.03 [-0.50, +0.56]	-0.43 [-0.75, -0.15]	-0.03 [-0.07, +0.00]
verify action on vs off	T2	-0.98 [-1.70, -0.41]	-1.45 [-2.14, -0.85]	-0.31 [-0.52, -0.13]
verify action on vs off	T3	-0.13 [-0.48, +0.20]	-1.32 [-1.92, -0.78]	-0.14 [-0.30, -0.01]
$n=20$ vs full-target cost model	T1	+0.00 [+0.00, +0.00]	+0.00 [-0.02, +0.03]	+0.00 [+0.00, +0.00]
$n=20$ vs full-target cost model	T2	+0.00 [+0.00, +0.00]	+0.00 [+0.00, +0.01]	+0.01 [+0.00, +0.02]
$n=20$ vs full-target cost model	T3	+0.00 [+0.00, +0.00]	-0.00 [-0.00, +0.00]	-0.00 [-0.01, +0.00]
$n=5$ vs source-only	T1	+0.15 [-0.12, +0.44]	+0.36 [+0.22, +0.52]	+0.00 [+0.00, +0.00]
$n=5$ vs source-only	T2	+0.05 [-0.03, +0.15]	+0.60 [+0.44, +0.76]	+0.52 [+0.30, +0.76]
$n=5$ vs source-only	T3	+0.00 [-0.04, +0.04]	+0.18 [+0.11, +0.25]	+0.17 [+0.08, +0.28]
$n=80$ vs source-only	T1	+0.15 [-0.12, +0.44]	+0.44 [+0.27, +0.63]	+0.00 [+0.00, +0.00]
$n=80$ vs source-only	T2	+0.05 [-0.03, +0.15]	+0.60 [+0.45, +0.77]	+0.53 [+0.30, +0.77]
$n=80$ vs source-only	T3	+0.00 [-0.04, +0.04]	+0.18 [+0.11, +0.25]	+0.17 [+0.08, +0.28]
calibrated vs fixed $k=1$	T1	+1.26 [+0.38, +2.36]	+2.31 [+1.77, +2.86]	+0.00 [+0.00, +0.00]
calibrated vs fixed $k=1$	T2	+0.27 [-0.12, +0.76]	+4.25 [+3.26, +5.35]	+0.54 [+0.32, +0.79]
calibrated vs fixed $k=1$	T3	+0.36 [-0.01, +0.82]	+4.58 [+3.64, +5.55]	+0.29 [+0.16, +0.44]

Table 9: Transfer matrix on SVAMP (held-out task family), as Table 4.

source→target	b viol. (%)	a viol. (%)	Δ viol. (pp)	Δ acc. (pp)	Δ J	first-attempt infeasible (%)
S→S (in-regime)	1.2	1.2	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	0.9
S→T1	0.4	0.5	+0.1 [+0.0, +0.1]	+0.1 [-0.2, +0.4]	+0.3	0.4
S→T2	0.2	0.4	+0.2 [+0.1, +0.4]	+0.0 [-0.1, +0.1]	+0.2	0.2
S→T3	0.4	0.5	+0.1 [+0.0, +0.1]	+0.0 [+0.0, +0.1]	+0.1	0.4
T1→S	46.2	42.1	-4.1 [-5.0, -3.3]	-0.2 [-0.4, -0.1]	-1.8	41.9
T1→T1 (in-regime)	3.1	3.1	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	1.9
T1→T2	29.4	25.8	-3.6 [-4.3, -3.0]	-0.1 [-0.3, +0.0]	-1.3	25.6
T1→T3	33.4	29.4	-4.0 [-4.8, -3.2]	-0.1 [-0.3, +0.0]	-1.4	28.9
T2→S	8.5	4.5	-4.0 [-4.8, -3.1]	-0.2 [-0.5, -0.0]	-3.2	3.1
T2→T1	0.9	1.8	+0.8 [+0.4, +1.4]	+0.6 [-0.1, +1.4]	+1.0	0.6
T2→T2 (in-regime)	2.6	2.6	+0.0 [-0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	0.9
T2→T3	3.1	2.5	-0.7 [-1.0, -0.4]	+0.0 [-0.1, +0.1]	-0.8	1.1
T3→S	2.6	2.4	-0.2 [-0.4, -0.1]	-0.0 [-0.1, +0.0]	-0.3	2.2
T3→T1	0.4	0.5	+0.0 [+0.0, +0.1]	+0.0 [-0.1, +0.1]	+0.1	0.4
T3→T2	0.8	0.9	+0.1 [+0.0, +0.2]	+0.1 [+0.0, +0.3]	+0.1	0.6
T3→T3 (in-regime)	1.0	1.0	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	0.8

Table 10: Confirmatory (fresh) transfer matrix, calibrated ($n=20$) minus source-only.

source→target	b viol. (%)	a viol. (%)	Δ viol. (pp)	Δ acc. (pp)	Δ J	first-attempt infeasible (%)
S→S (in-regime)	3.5	2.9	-0.6 [-0.7, -0.4]	-0.4 [-0.5, -0.2]	-3.4	2.0
S→T1	0.1	0.3	+0.2 [+0.1, +0.4]	+2.0 [+1.1, +2.8]	+3.9	0.0
S→T2	0.1	0.9	+0.8 [+0.6, +1.1]	+0.4 [+0.2, +0.7]	+3.9	0.0
S→T3	0.7	1.5	+0.8 [+0.6, +1.0]	+0.0 [-0.3, +0.3]	+1.3	0.4
T1→S	77.1	61.1	-16.0 [-17.1, -14.9]	-0.0 [-0.2, +0.1]	-7.2	61.0
T1→T1 (in-regime)	3.9	3.2	-0.6 [-0.9, -0.4]	-0.6 [-0.9, -0.4]	-1.7	2.4
T1→T2	59.4	43.3	-16.1 [-17.1, -15.0]	-0.8 [-1.2, -0.5]	-7.1	42.4
T1→T3	65.4	49.1	-16.3 [-17.4, -15.3]	-0.4 [-0.7, -0.2]	-8.0	48.7
T2→S	17.0	10.4	-6.6 [-7.4, -5.8]	-0.9 [-1.3, -0.6]	-9.6	9.5
T2→T1	0.2	0.5	+0.3 [+0.1, +0.4]	+1.9 [+1.3, +2.6]	+3.6	0.1
T2→T2 (in-regime)	3.8	3.4	-0.4 [-0.6, -0.2]	-0.3 [-0.4, -0.2]	-1.5	2.4
T2→T3	6.4	4.7	-1.8 [-2.2, -1.3]	-0.8 [-1.1, -0.5]	-4.3	3.7
T3→S	10.5	6.9	-3.6 [-4.2, -3.0]	-0.5 [-0.8, -0.3]	-5.8	6.2
T3→T1	0.1	0.3	+0.1 [+0.1, +0.2]	+1.3 [+0.7, +1.8]	+2.8	0.1
T3→T2	1.6	1.6	+0.1 [-0.1, +0.3]	-0.1 [-0.2, +0.1]	-0.0	1.0
T3→T3 (in-regime)	3.4	2.9	-0.5 [-0.7, -0.3]	-0.3 [-0.4, -0.1]	-1.7	2.1

Table 11: Fresh data: calibrated from the ORIGINAL container’s traces (*a*) minus source-only (*b*).

source→target	<i>b</i> viol. (%)	<i>a</i> viol. (%)	Δ viol. (pp)	Δ acc. (pp)	Δ J	first-attempt infeasible (%)
S→S (in-regime)	3.5	3.5	+0.0 [+0.0, +0.1]	+0.0 [+0.0, +0.0]	+0.1	2.0
S→T1	0.1	0.4	+0.3 [+0.1, +0.6]	+2.2 [+1.3, +3.1]	+4.2	0.0
S→T2	0.1	1.4	+1.3 [+0.9, +1.7]	+0.7 [+0.4, +1.1]	+5.8	0.0
S→T3	0.7	2.2	+1.4 [+1.0, +1.8]	+0.4 [+0.0, +0.7]	+4.0	0.4
T1→S	77.1	61.1	-16.0 [-17.2, -14.9]	-0.0 [-0.2, +0.1]	-7.2	61.0
T1→T1 (in-regime)	3.9	3.9	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.1]	+0.0	2.4
T1→T2	59.4	43.3	-16.1 [-17.1, -15.0]	-0.8 [-1.2, -0.5]	-6.5	42.4
T1→T3	65.4	49.2	-16.2 [-17.2, -15.2]	-0.4 [-0.7, -0.2]	-7.4	48.7
T2→S	17.0	11.0	-6.0 [-6.8, -5.2]	-0.6 [-0.9, -0.3]	-5.9	9.5
T2→T1	0.2	0.8	+0.5 [+0.3, +0.8]	+2.5 [+1.7, +3.3]	+4.3	0.1
T2→T2 (in-regime)	3.8	3.8	+0.0 [-0.0, +0.1]	+0.0 [-0.0, +0.0]	+0.1	2.4
T2→T3	6.4	5.2	-1.2 [-1.6, -0.8]	-0.4 [-0.6, -0.2]	-2.1	3.7
T3→S	10.5	7.5	-3.1 [-3.6, -2.5]	-0.3 [-0.5, -0.1]	-3.1	6.2
T3→T1	0.1	0.4	+0.3 [+0.1, +0.4]	+1.4 [+0.8, +2.0]	+3.2	0.1
T3→T2	1.6	1.9	+0.4 [+0.2, +0.6]	+0.1 [-0.0, +0.3]	+1.3	1.0
T3→T3 (in-regime)	3.4	3.4	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	2.1

Table 12: Fresh data, H2: calibrated controller without verify (*a*) minus calibrated budgeted-*k* (*b*).

source→target	<i>b</i> viol. (%)	<i>a</i> viol. (%)	Δ viol. (pp)	Δ acc. (pp)	Δ J	first-attempt infeasible (%)
S→S (in-regime)	2.6	2.9	+0.3 [+0.1, +0.5]	-0.2 [-0.6, +0.1]	-20.5	2.0
S→T1	0.3	0.3	-0.1 [-0.2, +0.0]	-1.4 [-2.2, -0.7]	-13.6	0.0
S→T2	0.5	0.9	+0.4 [+0.2, +0.6]	-0.6 [-1.2, -0.2]	-22.0	0.0
S→T3	1.1	1.6	+0.5 [+0.2, +0.7]	-0.6 [-1.1, -0.2]	-21.7	0.4
T1→S	61.1	61.1	+0.0 [-0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	61.0
T1→T1 (in-regime)	2.9	3.1	+0.2 [+0.1, +0.3]	-1.2 [-1.7, -0.6]	-4.6	2.4
T1→T2	42.7	42.8	+0.1 [-0.0, +0.3]	+0.0 [-0.0, +0.1]	+0.3	42.4
T1→T3	48.9	49.1	+0.2 [+0.1, +0.3]	-0.0 [-0.1, +0.0]	+0.3	48.7
T2→S	10.9	10.4	-0.5 [-0.7, -0.3]	-0.1 [-0.2, +0.0]	-17.5	9.5
T2→T1	0.8	0.6	-0.2 [-0.3, -0.1]	-1.1 [-1.9, -0.4]	-14.2	0.1
T2→T2 (in-regime)	3.8	3.4	-0.3 [-0.5, -0.2]	-0.4 [-0.7, -0.1]	-20.5	2.4
T2→T3	5.1	4.8	-0.3 [-0.4, -0.1]	-0.2 [-0.4, -0.0]	-19.4	3.7
T3→S	6.8	6.9	+0.1 [-0.1, +0.3]	-0.9 [-1.5, -0.4]	-27.0	6.2
T3→T1	0.4	0.3	-0.2 [-0.3, -0.0]	-4.7 [-6.3, -3.1]	-20.6	0.1
T3→T2	1.6	1.6	+0.1 [-0.1, +0.2]	-2.7 [-3.7, -1.8]	-32.2	1.0
T3→T3 (in-regime)	2.8	2.8	-0.0 [-0.2, +0.2]	-2.2 [-3.1, -1.4]	-31.0	2.1

Table 13: Round 2: budgeted-*k* calibrated minus budgeted-*k* source-only (held-out GSM8K, pooled).

source→target	<i>b</i> viol. (%)	<i>a</i> viol. (%)	Δ viol. (pp)	Δ acc. (pp)	Δ J	first-attempt infeasible (%)
S→S (in-regime)	3.5	3.5	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	2.6
S→T1	0.0	0.1	+0.1 [+0.0, +0.2]	+2.4 [+1.5, +3.3]	+8.9	0.0
S→T2	0.0	1.0	+1.0 [+0.7, +1.4]	+1.0 [+0.6, +1.4]	+10.0	0.0
S→T3	0.6	1.8	+1.2 [+0.8, +1.6]	+0.3 [-0.1, +0.8]	+7.3	0.4
T1→S	80.5	62.3	-18.3 [-19.5, -17.0]	-0.1 [-0.3, +0.2]	-9.6	62.2
T1→T1 (in-regime)	3.3	3.4	+0.0 [+0.0, +0.1]	+0.0 [+0.0, +0.0]	+0.0	2.3
T1→T2	66.4	44.3	-22.1 [-23.3, -20.8]	-0.6 [-1.0, -0.2]	-10.8	43.7
T1→T3	71.8	50.2	-21.6 [-22.8, -20.4]	+0.0 [-0.3, +0.3]	-11.1	50.0
T2→S	22.8	13.6	-9.2 [-10.3, -8.1]	-0.5 [-1.0, -0.1]	-9.8	11.4
T2→T1	0.0	1.1	+1.1 [+0.7, +1.4]	+2.1 [+1.4, +2.9]	+7.3	0.0
T2→T2 (in-regime)	5.0	5.1	+0.1 [+0.0, +0.3]	+0.0 [+0.0, +0.0]	+0.1	2.2
T2→T3	8.5	6.3	-2.2 [-2.7, -1.6]	-0.1 [-0.4, +0.1]	-2.7	3.8
T3→S	12.2	8.3	-3.9 [-4.6, -3.1]	-0.2 [-0.6, +0.1]	-9.0	7.4
T3→T1	0.0	0.3	+0.3 [+0.1, +0.5]	+2.3 [+1.5, +3.3]	+8.7	0.0
T3→T2	1.6	2.4	+0.8 [+0.4, +1.1]	+0.3 [+0.1, +0.5]	+3.2	1.1
T3→T3 (in-regime)	3.7	3.7	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	2.3

Table 14: Round 2: predictive-reserve controller without verify (*a*) minus calibrated budgeted-*k* (*b*), held-out GSM8K, pooled. Post-hoc comparison (see text).

source→target	<i>b</i> viol. (%)	<i>a</i> viol. (%)	Δ viol. (pp)	Δ acc. (pp)	Δ J	first-attempt infeasible (%)
S→S (in-regime)	3.5	4.6	+1.0 [+0.6, +1.5]	-0.1 [-0.7, +0.5]	-19.0	2.6
S→T1	0.1	0.0	-0.0 [-0.2, +0.1]	-0.7 [-1.5, +0.0]	-13.0	0.0
S→T2	1.0	1.8	+0.8 [+0.4, +1.3]	-0.1 [-0.5, +0.2]	-21.6	0.0
S→T3	1.8	2.9	+1.1 [+0.6, +1.6]	-0.1 [-0.3, +0.2]	-20.3	0.4
T1→S	62.3	62.3	+0.0 [-0.0, +0.1]	+0.0 [+0.0, +0.0]	+0.0	62.2
T1→T1 (in-regime)	3.4	4.0	+0.6 [+0.3, +1.0]	-1.0 [-1.8, -0.3]	-5.3	2.3
T1→T2	44.3	44.3	+0.0 [-0.3, +0.3]	+0.0 [+0.0, +0.0]	+0.4	43.7
T1→T3	50.2	50.5	+0.3 [+0.1, +0.6]	+0.0 [+0.0, +0.0]	+0.5	50.0
T2→S	13.6	13.0	-0.6 [-1.0, -0.3]	-0.1 [-0.4, +0.1]	-16.5	11.4
T2→T1	1.1	0.6	-0.5 [-0.8, -0.2]	-0.3 [-0.9, +0.3]	-13.9	0.0
T2→T2 (in-regime)	5.1	4.3	-0.8 [-1.2, -0.5]	-0.0 [-0.3, +0.2]	-20.1	2.2
T2→T3	6.3	5.7	-0.6 [-0.9, -0.3]	-0.4 [-0.6, -0.1]	-19.4	3.8
T3→S	8.3	8.5	+0.2 [-0.3, +0.6]	-1.5 [-2.3, -0.8]	-26.7	7.4
T3→T1	0.3	0.3	-0.0 [-0.3, +0.2]	-4.9 [-7.0, -3.0]	-21.7	0.0
T3→T2	2.4	2.4	+0.1 [-0.3, +0.5]	-1.9 [-2.9, -1.0]	-32.7	1.1
T3→T3 (in-regime)	3.7	3.5	-0.2 [-0.7, +0.2]	-1.8 [-2.6, -1.0]	-31.6	2.3

Table 15: Round 2 transfer matrix (predictive-reserve controller, calibrated minus source-only), held-out GSM8K.

source→target	<i>b</i> viol. (%)	<i>a</i> viol. (%)	Δ viol. (pp)	Δ acc. (pp)	Δ J	first-attempt infeasible (%)
S→S (in-regime)	4.4	4.4	+0.0 [+0.0, +0.0]	-0.0 [-0.0, +0.0]	+0.0	2.6
S→T1	0.0	0.1	+0.1 [+0.0, +0.2]	+1.9 [+1.1, +2.9]	+5.0	0.0
S→T2	0.1	1.6	+1.6 [+1.1, +2.0]	+0.4 [+0.1, +0.7]	+6.1	0.0
S→T3	1.0	2.8	+1.8 [+1.3, +2.3]	+0.4 [+0.1, +0.8]	+4.5	0.4
T1→S	77.0	62.3	-14.7 [-15.9, -13.5]	-0.0 [-0.2, +0.1]	-6.8	62.2
T1→T1 (in-regime)	4.2	4.2	+0.0 [+0.0, +0.1]	+0.0 [+0.0, +0.0]	+0.0	2.3
T1→T2	62.3	44.7	-17.6 [-18.8, -16.5]	-0.4 [-0.8, -0.1]	-7.1	43.7
T1→T3	67.1	50.4	-16.7 [-17.9, -15.6]	+0.0 [-0.3, +0.4]	-7.5	50.0
T2→S	18.3	12.9	-5.4 [-6.2, -4.6]	-0.3 [-0.7, +0.0]	-5.6	11.4
T2→T1	0.0	0.6	+0.6 [+0.3, +0.9]	+2.0 [+1.2, +3.0]	+5.1	0.0
T2→T2 (in-regime)	4.0	4.1	+0.0 [-0.0, +0.1]	+0.0 [-0.0, +0.0]	+0.0	2.2
T2→T3	6.8	5.5	-1.4 [-1.8, -0.9]	-0.2 [-0.5, +0.0]	-2.2	3.8
T3→S	12.6	8.6	-4.0 [-4.6, -3.3]	-0.2 [-0.5, -0.0]	-3.8	7.4
T3→T1	0.0	0.3	+0.3 [+0.1, +0.6]	+0.9 [+0.4, +1.5]	+3.4	0.0
T3→T2	1.9	2.6	+0.7 [+0.4, +1.0]	+0.2 [-0.0, +0.3]	+1.4	1.1
T3→T3 (in-regime)	3.6	3.6	+0.0 [+0.0, +0.0]	+0.0 [+0.0, +0.0]	+0.0	2.3

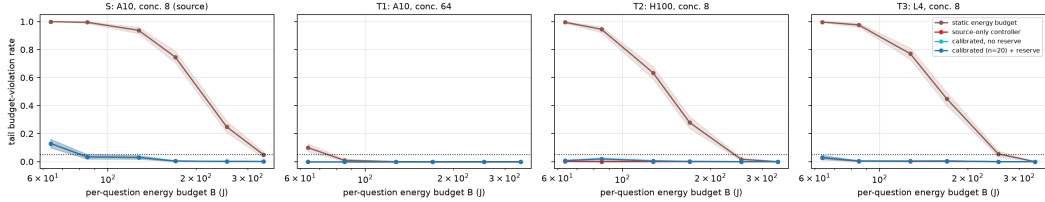


Figure 7: Budget-violation rate vs. budget (held-out GSM8K), with 95% CIs; dotted line: 5%.

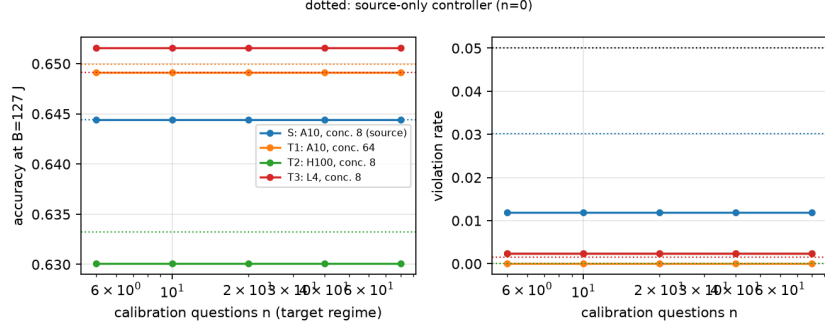


Figure 8: Calibration-set size n vs. accuracy and violation at $B = 127.4$ J (gain seed 0, five draws).

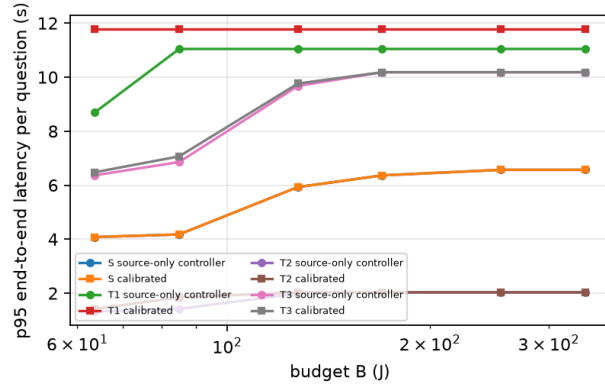


Figure 9: p95 end-to-end latency per question vs. budget.

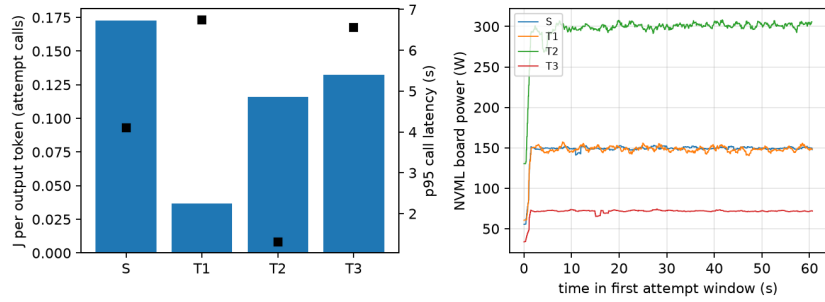


Figure 10: Left: J per output token and p95 call latency per regime. Right: NVML board power during the first minute of the first attempt window.

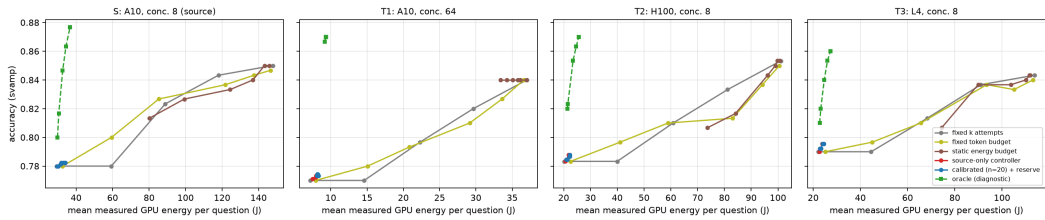


Figure 11: Accuracy-energy frontier on SVAMP (held-out task family).