SGM: A Statistical Gödel Machine for Risk-Controlled Recursive Self-Modification

Anonymous authors
Paper under double-blind review

Abstract

Recursive self-modification is increasingly central in AutoML, neural architecture search, and adaptive optimization, yet no existing framework ensures that such changes are made safely. Gödel machines offer a principled safeguard by requiring formal proofs of improvement before rewriting code; however, such proofs are unattainable in stochastic, high-dimensional settings. We introduce the *Statistical Gödel Machine (SGM)*, the first *statistical safety layer for recursive edits*. SGM replaces proof-based requirements with statistical confidence tests (e-values, Hoeffding bounds), admitting a modification only when superiority is certified at a chosen confidence level, while allocating a global error budget to bound cumulative risk across rounds. We also propose *Confirm-Triggered Harmonic Spending (CTHS)*, which indexes spending by confirmation events rather than rounds, concentrating the error budget on promising edits while preserving familywise validity. Experiments across supervised learning, reinforcement learning, and black-box optimization validate this role: SGM certifies genuine gains on CIFAR-100, rejects spurious improvement on ImageNet-100, and demonstrates robustness on RL and optimization benchmarks. Together, these results position SGM as foundational infrastructure for continual, risk-aware self-modification in learning systems. Code is available at: `https://github.com/gravitywavelet/sgm-anon`.

1 Introduction

Recursive self-modification has often been discussed as a cornerstone for building continually improving ML systems (Yampolskiy, 2015). Modern ML already hints at this trend: reinforcement learning agents tune hyperparameters online, AutoML loops search over training recipes, and optimization pipelines reconfigure code and settings during runs. Yet these procedures often adopt changes on the basis of noisy gains, creating the risk of harmful edits – modifications that seems beneficial in finite trials but ultimately degrade true performance. Such risks are especially concerning in high-stakes scientific domains such as drug design, protein engineering, or climate modeling, where spurious gains can misdirect costly pipelines.

Gödel machines (Schmidhuber, 2007) offer a conceptually clean answer: an agent rewrites its code only when it can *prove* the rewrite increases expected utility. But in stochastic, high-dimensional ML, such formal proofs are unattainable. At the other extreme, practical AutoML and RL systems adopt edits using heuristics such as rolling averages, best-of-seeds, or bandit rules, which lack guarantees and may silently accumulate regressions. This gap motivates our question:

Can we provide a principled safety layer for recursive edits, ensuring that self-modification proceeds only when supported by rigorous statistical evidence?

We introduce the *Statistical Gödel Machine (SGM)*, which establishes the first *statistical safety layer for recursive self-modification*. Instead of demanding logical proofs, SGM admits a modification only when statistical certificates certify superiority at a chosen confidence level. To remain safe across many rounds, SGM allocates a global error budget using union-bound splits or anytime spending rules (e.g., e-values), bounding the probability of ever adopting a harmful change. Importantly, SGM is not designed to generate stronger proposals, but to serve as a *risk-control framework that can wrap around arbitrary proposers*, consistently filtering noise while preserving genuine progress.

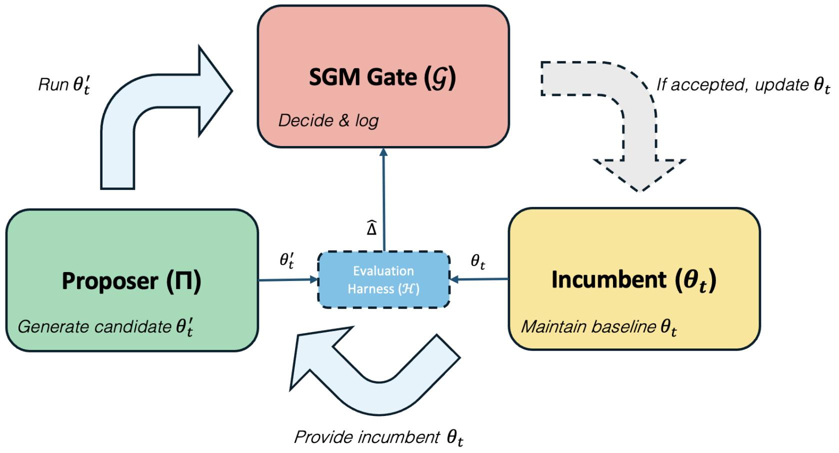
Run θ_t'
SGM Gate ($\mathcal{G}$)
Decide & log
If accepted, update θ_t
$\hat{\Delta}$
Proposer (Π)
Generate candidate θ_t'
θ_t'
Evaluation Harness ($\mathcal{H}$)
θ_t
Incumbent (θ_t)
Maintain baseline θ_t
Provide incumbent θ_t

Figure 1: SGM architecture: At each round t, the **Proposer** (Π) generates a candidate θ_t', which is compared to the current **Incumbent** (θ_t) by the **Evaluation Harness** ($\mathcal{H}$). The **SGM Gate** ($\mathcal{G}$) then applies statistical tests to certify or reject the edit, ensuring risk-controlled acceptance with bounded error probability. If certified, the incumbent is updated; otherwise, the system remains unchanged.

Unlike standard sequential testing or online false discovery rate (FDR) methods, SGM governs *irreversible commits*: each accepted edit rewrites the incumbent and persists into future rounds, requiring error control across an open-ended sequence.

We evaluate SGM across reinforcement learning, black-box optimization, and supervised learning. On CIFAR-100, SGM certified a genuine $+5.5$pp gain under a 30-seed stress test, while on ImageNet-100 it correctly rejected a seemingly promising edit that failed confirmation. These results highlight SGM's role as a reusable risk-control layer for self-improving ML pipelines.

Contributions

This paper makes three contributions:

- **Statistical safety for recursive edits.** We introduce SGM, the first framework to replace Gödel's proof-based requirement with PAC-style statistical tests, enabling safe self-modification in noisy, high-dimensional ML.
- **Event-triggered cumulative risk control.** We propose confirmation-triggered harmonic spending (CTHS), which concentrates the error budget on rounds that escalate to confirmation, improving power on promising edits while preserving familywise validity.
- **Cross-domain validation.** We demonstrate SGM across supervised learning (CIFAR-100, ImageNet-100), reinforcement learning, and black-box optimization, showing both certified gains and principled rejections.

2 Related Work

Gödel machines and self-referential improvement. Gödel machines formalize fully self-referential agents that rewrite their own code once a proof guarantees higher expected utility (Schmidhuber, 2007). While conceptually elegant, such proofs are unattainable in stochastic, high-dimensional ML. SGM retains the same outer-loop self-edit structure but *relaxes proof obligations into PAC-style statistical guarantees with cumulative error control*, yielding a tractable analogue for modern pipelines.

Statistical testing and adaptive error control. SGM builds on concentration inequalities such as Hoeffding bounds (Hoeffding, 1963) and variance-adaptive alternatives like empirical Bernstein bounds (Maurer & Pontil, 2009). To ensure long-run safety, it leverages either conservative union-bound schedules or adaptive anytime spending rules such as α-investing and e-values (Foster &

Stine, 2008; Howard et al., 2020; Waudby-Smith & Ramdas, 2024). Prior work in sequential analysis and adaptive testing (e.g., LIL bounds, mixture SPRTs) regulates error rates under adaptivity, but has not been applied to recursive self-modification where accepted edits persist.

Safe RL, AutoML, and adaptive selection. Safe RL methods constrain expected cost or risk through CMDPs (Altman, 2021), trust-region or Lagrangian techniques, or shielded policies (Achiam et al., 2017; Tessler et al., 2018; Garcıa & Fernández, 2015). These safeguard *policy execution* during learning, whereas SGM safeguards *system-level commits* that permanently alter the learner. Similarly, Bayesian optimization (BO) and AutoML frameworks (Shahriari et al., 2015; Li et al., 2018; Falkner et al., 2018; Jaderberg et al., 2017) improve models by balancing exploration and exploitation, but typically adopt candidates heuristically based on noisy validation estimates. Recent work on risk-aware BO introduces conservative acquisition rules, yet still governs *trial allocation* rather than irreversible adoption. By contrast, SGM is complementary to AutoML: it does not propose candidates more efficiently, but acts as a *drop-in gate* that certifies any proposed edit with explicit statistical guarantees before it is committed.

Adaptivity, online FDR, and A/B testing. Repeated adaptivity can invalidate p-values and confidence intervals (Armitage et al., 1969). Methods such as selective inference, knockoffs, and online FDR (Fithian et al., 2014; Barber & Candès, 2015; Ramdas et al., 2018) address this in external evaluations, while adaptive A/B testing frameworks (Johari et al., 2022) mitigate bias under repeated peeking. These approaches regulate *temporary* error rates, whereas SGM governs *irreversible* commits inside a self-modifying system. This persistence motivates familywise error control (FWER) rather than FDR, since a single harmful commit can permanently degrade performance.

Program synthesis and self-modifying ML systems. Meta-learning, NAS, and program synthesis (Zoph & Le, 2016; Real et al., 2019; Gaunt et al., 2016) demonstrate empirical self-improvement but rely on heuristic adoption rules. Proof-carrying code and certified optimization passes in PL/verification (Necula, 1997; Leroy, 2009) demand logical certificates. SGM occupies a middle ground: edits need not be logically proven, but must satisfy statistically certified advantages with controlled cumulative error—bridging principled safety with data-driven ML.

3 Methods

3.1 SGM Gate (Interface)

Definition 1 (SGM Gate). *At outer round t, a proposer Π maps the transcript $\mathcal{T}_{t-1}$ (past proposals, outcomes, and decisions) to a candidate θ'_t, while a harness $\mathcal{H}$ produces paired outcomes comparing incumbent θ_t and proposal. The* SGM Gate *$\mathcal{G}$ outputs a decision $D_t \in \{\text{ACCEPT}, \text{REJECT}\}$ and a statistical certificate $C_t(\delta_t)$, while managing a global error budget δ.*

The gate enforces two guarantees:

- ***Per-edit safety.** For each round t,*
$$\Pr(\textit{harmful accept at } t) \;\le\; \delta_t.$$
- ***Cumulative safety.** For any horizon T',*
$$\Pr\big(\exists t \le T' : \textit{harmful accept}\big) \;\le\; \delta,$$
using either fixed allocation ($\delta_t = \delta/T$) or an anytime spending rule (e.g. e-values, α-investing).

3.2 Statistical Guarantees

The SGM gate provides rigorous, distribution-free guarantees for deciding whether to accept a proposed configuration. All tests operate on *bounded paired differences* $\Delta_i \in [a, b]$, oriented so that $\Delta_i > 0$ denotes improvement (accuracy or reward gains, or negated loss/error). We normalize by $R = \max\{|a|, |b|\}$ and define the mean normalized improvement

$$\bar{\Delta} = \frac{1}{n}\sum_{i=1}^{n} \frac{\Delta_i}{R} \in [-1, 1].$$

Error allocation. To control familywise error across B proposal rounds, we allocate the global tolerance δ using a harmonic schedule:

$$\delta_t = \frac{\delta}{tH_B}, \qquad H_B = \sum_{i=1}^{B} \frac{1}{i}.$$

This ensures $\sum_{t=1}^{B} \delta_t = \delta$, bounding the probability of *any* false acceptance across all rounds by δ.

3.2.1 Fixed-δ Acceptance via Hoeffding

Hoeffding acceptance rule. Given paired improvements $X_i \in [a, b]$, the empirical mean $\hat{\mu} = \frac{1}{n}\sum_i X_i$ admits a one-sided lower confidence bound

$$\mathrm{LCB}_{1-\delta} = \hat{\mu} - (b-a)\sqrt{\tfrac{1}{2n} \ln \tfrac{1}{\delta}}.$$

We accept a proposal only if $\mathrm{LCB}_{1-\delta} > 0$.

This follows from Hoeffding's inequality, which guarantees that

$$\Pr\big(\mu < \mathrm{LCB}_{1-\delta}\big) \;\le\; \delta.$$

Theorem 1 (Union-Bound Guarantee)**.** *If each round is tested with $\delta_t = \delta/T$, then with probability at least $1-\delta$ no harmful modification is accepted across T rounds:*

$$\Pr\Big(\exists t \le T : \mu_t < 0 \,\wedge\, \textit{accepted}\Big) \;\le\; \delta.$$

This rule follows directly from Hoeffding's inequality (Appendix A.1).

3.2.2 Variance-Adaptive Acceptance (Empirical Bernstein)

Hoeffding's bound ignores variance, leading to loose thresholds. The empirical Bernstein inequality (Maurer & Pontil, 2009) adapts to observed variance:

$$\Pr\left(\mu < \hat{\mu} - \sqrt{\tfrac{2\hat{\sigma}^2 \ln(3/\delta)}{n}} - \tfrac{3(b-a)\ln(3/\delta)}{n}\right) \le \delta, \tag{1}$$

where $\hat{\sigma}^2$ is the empirical variance. This yields tighter confidence intervals in low-variance regimes (e.g., deterministic optimization), while retaining PAC guarantees.

3.2.3 Anytime Acceptance via E-Values

Unlike the fixed-n tests above, e-values provide an *anytime-valid guarantee* (Howard et al., 2020), enabling sequential testing over an unbounded horizon. At round t, we compare the incumbent θ_t with the proposal θ'_t using paired differences $\Delta_{t,i} \in [a, b]$. Define the normalized variables $X_{t,i} = \Delta_{t,i}/R \in [-1, 1]$, where $R = \max\{|a|, |b|\}$.

Let $\lambda_{t,i} \in [0, 1]$ be a predictable choice (based only on past data, not the current sample) that determines the betting fraction: small λ yields conservative updates, while large λ increases sensitivity but also variance. In our implementation, we fix $\lambda_{t,i} = 1$, corresponding to staking the full fraction on each observation. Given this choice, we define the per-sample e-value and cumulative updates as

$$e_{t,i} = 1 + \lambda_{t,i} X_{t,i}, \tag{2}$$

$$E_t = \prod_{i=1}^{n_t} e_{t,i}, \tag{3}$$

$$W_t = W_{t-1} E_t, \quad W_0 = 1. \tag{4}$$

Here $e_{t,i}$ is the contribution from a single paired sample, E_t aggregates evidence across all n_t samples in round t, and W_t is the running wealth across rounds.

By construction, $\mathbb{E}[e_{t,i} \mid \mathcal{F}_{t,i-1}] \le 1$ (Ramdas et al., 2018), where $\mathcal{F}_{t,i-1}$ denotes the information available up to sample $i-1$ in round t (i.e., all past outcomes, but not the current one). Thus, the wealth process $\{W_t\}$ is a nonnegative supermartingale, and Ville's inequality gives

$$\Pr\big(\sup_{t \ge 1} W_t \ge 1/\delta\big) \le \delta.$$

Acceptance rule: adopt θ'_t once $W_t \geq 1/\delta$.

Theorem 2 (Anytime Control via E-Values)**.** *With bounded differences and predictable $\lambda_{t,i}$, the wealth process $\{W_t\}$ is a nonnegative supermartingale. Thus, the stopping time $\tau = \inf\{t : W_t \geq 1/\delta\}$ controls false acceptance:*

$$\Pr(\exists t : \textit{accept at round } t \textit{ when } \mu_t \leq 0) \leq \delta.$$

Practical guidance. Each acceptance mode offers distinct strengths:

- **Hoeffding (fixed-δ).** Simple, conservative, and variance-agnostic; suited to small n with fixed budgets under a union-bound split.
- **Empirical Bernstein (variance-adaptive).** Tighter in low-variance regimes, but overly conservative under high variance or heavy tails.
- **E-values (anytime).** Default in our experiments; support early stopping and mini-batching without pre-allocating δ.

In implementation, we instantiate the gate with e-values by default, using Hoeffding or empirical Bernstein as drop-in alternatives for fixed-budget or low-variance settings.

Summary. These acceptance rules ensure that the SGM gate only admits proposals with statistically certified improvement, while bounding the global error rate across multiple iterations.

Novelty. While Hoeffding bounds, empirical Bernstein bounds, and e-values are well-established in statistical learning, our novelty lies in repurposing these tools as *gates for recursive self-modification*. Rather than serving as external evaluation techniques, we reinterpret them as internal contracts that decide whether a proposed edit is permanently adopted. This shift from regulating temporary errors to governing irreversible updates differentiates SGM from classical adaptive testing and provides, to our knowledge, the first statistical framework explicitly designed for continual self-editing processes.

3.3 Algorithm

Algorithm 1 SGM outer loop with certified acceptance.

```
Require: Initial config $\theta_0$; proposer $\Pi$; harness $\mathcal{H}$; max rounds $T$; global $\delta$
Ensure: Final config $\theta^\star$; registry $\mathcal{R}$
$\theta \leftarrow \theta_0$;   $\mathcal{R} \leftarrow \{(\theta_0, \texttt{baseline})\}$;   $W \leftarrow 1$
for $t = 1$ to $T$ do
    $\Theta_{\text{cand}} \leftarrow$ PROPOSE$(\Pi, \theta, \mathcal{R})$
    for all $\theta' \in \Theta_{\text{cand}}$ do
        $\{\Delta_i\}_{i=1}^n \leftarrow$ PAIREDEVALUATE$(\mathcal{H}, \theta, \theta')$
        $(\text{LCB}, W) \leftarrow$ CERTIFY$(\{\Delta_i\}, \delta, W)$        ▷ Applies tests from Sec. 3.2
        if LCB $> 0$ or $W \geq 1/\delta$ then
            accept $\theta'$; update $\theta, \mathcal{R}$
        else
            reject
return $(\theta^\star, \mathcal{R})$
```

4 Experiment

We evaluate SGM across supervised learning, reinforcement learning, and optimization tasks. Unless otherwise stated, we use paired seeds to estimate per-proposal improvements Δ_s and apply the decision rules from Sec. 3.2. For supervised learning (CIFAR, ImageNet), proposals undergo a two-stage screening/confirmation protocol (few seeds × short epochs, then many seeds × longer epochs if promising). Full hardware and compute details appear in Appendix B.

4.1 Experiment 1: CIFAR-100—CTHS and Deep-Learning Stress Test

Setup. We evaluate SGM on CIFAR-100, a high-variance benchmark, using a paired-seeds protocol: for each seed s, we train both the incumbent θ and the proposal θ', recording the paired difference Δ_s in percentage points (pp). Screening uses 4–6 seeds for 3–20 epochs, while promising candidates are escalated to confirmation with 12–30 seeds for 8–60 epochs.[1] Proposals mutate standard hyperparameters such as weight decay, EMA decay, and label smoothing. The safeguard is configured with δ=0.1, $r_{\max}$=1.0, and a heuristic screening trigger of 0.4 pp.

Part A: Confirm-Triggered Harmonic Spending (CTHS). To directly evaluate statistical power, we design a controlled *power analysis* experiment. At the confirmation stage only, we add a fixed offset of +4.0pp to the proposal's measured accuracy. This synthetic gain ensures that the proposal is genuinely superior at confirmation while leaving screening unchanged. Such controlled injections are common in power analysis and allow us to isolate the sensitivity of statistical schedules without conflating with real proposal noise.

We compare the standard harmonic schedule, which allocates δ_t by round index t, against *Confirm-Triggered Harmonic Spending (CTHS)*, which allocates by the k-th confirmation event. CTHS successfully certifies the improvement on its very first confirmation (round 1), then correctly rejects later noisy positives (rounds 5/6), spending $0.0748 < \delta = 0.10$. Harmonic, by contrast, does not encounter the gain until later confirmations (rounds 3–6), where its per-round δ_t is smaller, and thus makes no accepts, spending only 0.0388.

This result demonstrates that CTHS concentrates statistical power on the earliest promising event, thereby detecting genuine gains more effectively while still respecting the global error budget.

Table 1: CIFAR-100 synthetic power analysis (+4.0pp at confirmation). CTHS certifies the improvement early, while harmonic fails to accept the same gain due to smaller per-round δ_t.

Schedule	Conf. rounds	Total spend	Accepts	Outcome
CTHS	1, 5, 6	0.0748	**1**	Early accept; later rejections
Harmonic	3, 4, 5, 6	0.0388	0	Later confirms, lower δ_t

Part B: CIFAR-10 Sanity Check. Before turning to the higher-variance CIFAR-100 benchmark, we ran a lightweight validation on CIFAR-10. The baseline used SGD with batch size 768, while the proposal reduced the batch size to 64. Across 25–30 seeds, the safeguard consistently certified a modest but reliable gain (85.5% → 87.9%, $\text{LCB} > 0$), leading to acceptance. This simple test confirms that SGM can reliably detect small, consistent improvements in image classification.

Part C: Real Proposals on CIFAR-100 (Deep Learning Stress Test).

We next evaluate SGM on *real* hyperparameter proposals for CIFAR-100, a challenging high-variance benchmark. The goal is to test whether SGM can certify genuine improvements while filtering out noisy or misleading proposals.

Table 2: CIFAR-100 stress test with actual hyperparameter proposals (reporting incumbent and proposal accuracy %). Only iteration 6 achieves a certified gain under 30-seed confirmation.

Iter	Proposal Change(s)	Seeds	Inc. Acc.(%)	Prop. Acc.(%)	$\bar{\Delta}$ (pp)	$\text{LCB}_{1-\delta}$	Decision
1–5	`lr`, `ema`, `warmup` tweaks	6	56.05	[56.19,56.31]	$\leq +0.25$	< 0	Reject

6	`weight_decay=0.001`
	`ema_decay=0.99`
		30	56.05	61.56	+5.51	+0.31	**Accept**

7–10	`warmup`, `weight_decay` variations	6–30	57.45	[55.88,57.28]	$[-1.57, -0.40]$	< 0	Reject

Interpretation. Across 10 iterations, only iteration 6 (`weight_decay=0.001`, `ema_decay=0.99`) passes 30-seed, 60-epoch confirmation, improving accuracy from 56.05% to 61.56% (+5.51pp, $\text{LCB} = +0.31$). Screening uses a heuristic escalation rule, triggering confirmation once the mean improvement $\bar{\Delta}$ exceeds 0.8, while final acceptance is decided solely

[1] We use disjoint seed pools between screening and confirmation.

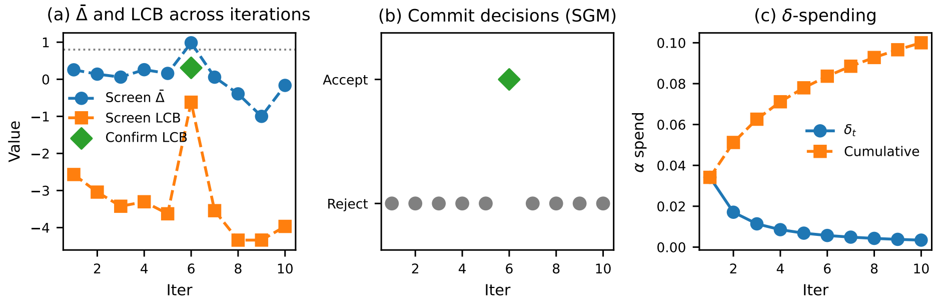

Figure 2: CIFAR-100 stress test under SGM. Only iteration 6 passes 30-seed confirmation, leading to acceptance. (a) Screening $\bar{\Delta}$ and LCB across iterations, with the dashed line indicating the escalation threshold. (b) Commit decisions under SGM: only iteration 6 is accepted. (c) δ-spending per iteration and cumulative total.

by confirm-stage LCBs. All other proposals either failed to escalate or were rejected despite positive screening means. Thus, SGM certifies only genuine gains while blocking noisy regressions, acting as a reliable statistical safeguard for deep learning pipelines. Moreover, the δ-spending curve shows this selectivity is achieved while respecting the global error budget (0.1).

Takeaway. On CIFAR-100, CTHS (Part A) shows stronger power by concentrating budget on early confirmations, while the CIFAR-10 sanity check (Part B) demonstrates that SGM reliably detects modest improvements. The CIFAR-100 stress test (Part C) confirms that SGM is conservative against noise yet able to certify genuine deep learning gains, establishing it as both statistically powerful and practically reliable for supervised learning.

4.2 Experiment 2: ImageNet-100 (Mid-Scale Test of Safety)

Setup. We extended the CIFAR-100 protocol (Experiment 1) to ImageNet-100 using DeiT-S/224, AdamW, cosine schedule, and EMA disabled. Each proposal was first screened with 4 seeds for 50 epochs and, if promising, escalated to confirmation with 12 seeds for 120 epochs. The gate used $\delta = 0.1$, $r_{\max} = 0.5$, and a screening threshold of 0.4pp.

Results. Screening suggested modest gains (+2.9pp for mixup $=$ 0.1), but all were overturned during confirmation. At iteration 6, the incumbent achieved 76.65% vs. 72.62% for the proposal ($\bar{\Delta} = -4.03$pp, LCB$= -1.91$), leading to rejection. Thus, no proposal achieved a certified gain on ImageNet-100.

Table 3: Experiment 2 (ImageNet-100). Screening (4 seeds, 50 epochs) vs. confirmation (12 seeds, 120 epochs). All proposals were rejected under confirmation.

Iter	Proposal	Seeds	Inc. Acc. (%)	Prop. Acc. (%)	$\bar{\Delta}$ (pp)	LCB$_{1-\delta}$	Decision
6	mixup=0.1, cutmix=0.0	12	76.65	72.62	-4.03	-1.91	Reject

4.3 Experiment 3: RL Safety Checks (CartPole and LunarLander)

Setup. We next test SGM in reinforcement learning tasks, focusing on its role as a safety filter rather than an optimizer. Both tasks use PPO with default hyperparameters. We run with $\delta = 0.1$ and budgets $B = 8$ (CartPole-v1) and $B = 3$ (LunarLander-v2). Paired seeds are used throughout.

CartPole-v1 (safety at saturation). Here the baseline PPO agent already solves the environment ($r_{\max} = 500$). We trained with 19 random seeds. Across 8 proposals, candidate modifications underperformed the strong incumbent (mean return 447.7 ± 54.5 vs. baseline 493.4 ± 12.6). The mean improvement was negative ($\bar{\Delta} = -45.7$) with LCB $[-0.95, -0.79]$, leading the safeguard to reject all proposals (Table 4, Fig. 3). This confirms the desired property: when the incumbent is near-optimal, SGM reliably blocks regressions.

Table 4: CartPole-v1: safety demo with 19 seeds, B=8, δ=0.1.

Config	n	Mean ± SD	#Props	Mean improv. ($\bar{\Delta}$)	LCB$_{1-\delta}$
Baseline	19	493.4 ± 12.6	–	–	–
Proposals	152	447.7 ± 54.5	8	-45.7	$[-0.95, -0.79]$

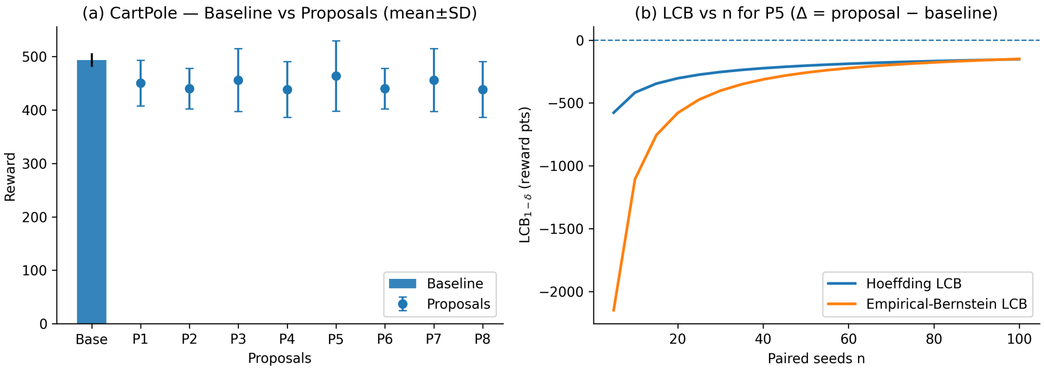

Figure 3: Ex1 (CartPole-v1, safety demo). Baseline vs. proposals: mean return with 95% CIs across 19 seeds. All proposals underperform baseline; no acceptance triggered by the gate.

LunarLander-v2 (safety under high variance). This environment is far noisier (baseline mean reward -479.5 ± 271.1). With $n = 19$ seeds and $B = 3$, the safeguard accepted one configuration that warm-started training, yielding a large gain of $\hat{\Delta} = +513.2 \pm 306.9$ and a certified lower bound $LCB_{1-\delta} = +0.04$ (Table 5, Fig. 4). Despite variance $\approx$ 307, the gate still identified and certified a genuine improvement.

Table 5: LunarLander-v2: n=19, $B = 3$, $\delta = 0.1$, $r_{\max} = 600$.

Config	n	Inc. mean ± SD	Prop. mean ± SD	$\bar{\Delta}$	LCB$_{1-\delta}$	Decision
Baseline	19	-479.5 ± 271.1	–	–	–	–
Proposal	19	–	33.7 ± 306.9	$+513.2$	$+0.04$	**Accept**

Takeaway. Together, these safety checks show both sides of SGM in RL: it reliably blocks regressions once an environment is saturated (CartPole), and it can still admit genuine gains even under extreme stochasticity (LunarLander). This balance of conservatism and sensitivity is central to SGM's role as a safety mechanism.

4.4 Experiment 4: Rastrigin20 (Optimization Stress Test)

Setup. To evaluate SGM in a nearly deterministic regime, we used the 20-dimensional Rastrigin function, a standard black-box optimization benchmark with many local minima (Hansen et al., 2010; 2021). The baseline optimizer was CMA-ES (Hansen et al., 2003) with step-size $\sigma = 0.5$ and population size 16. Proposals modified a single hyperparameter (e.g., reducing σ). Each evaluation used n = 80–100 random seeds, a budget of 2000 evaluations per seed, and confidence $\delta \in [0.30, 0.35]$.

Results. With 80–100 seeds, proposals showed only micro-improvements ($\bar{\Delta} \approx -0.55$). SGM certified cases where the lower confidence bound crossed zero, while rejecting others. This illustrates conservative behavior: SGM blocks spurious fluctuations but can admit genuine micro-gains.

Takeaway. Alongside Experiments 4.1 (CIFAR-100) and 4.3 (RL), Rastrigin20 demonstrates SGM's robustness across extremes: stochastic deep learning, high-variance RL, and nearly de-

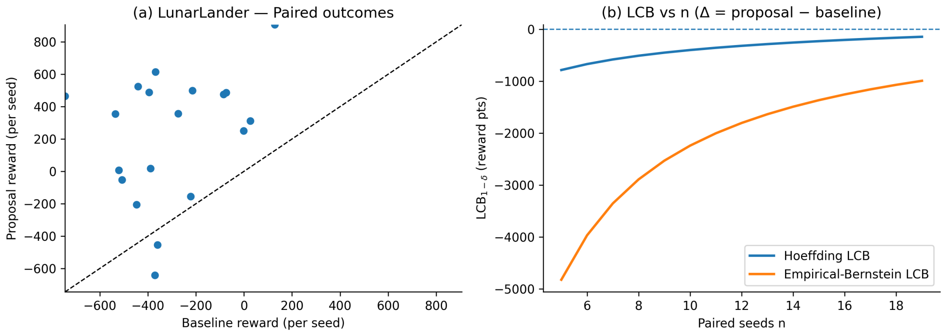

Figure 4: Experiment 2 (LunarLander-v2). (**a**) Paired *per-seed* returns: each dot is one random seed; the diagonal indicates parity (proposal = baseline). (**b**) Lower confidence bound on the improvement Δ = proposal − baseline as a function of paired seeds n. We plot $\mathrm{LCB}_{1-\delta}$ under Hoeffding (solid blue) and empirical-Bernstein (solid orange). A proposal is certified once $\mathrm{LCB}_{1-\delta} > 0$; for the accepted configuration, this occurs at $n = 19$. In this high-variance regime, Hoeffding is tighter because empirical-Bernstein over-penalizes variance.

Table 6: Rastrigin20 ($d = 20$). Lower f is better. SGM certifies only micro-gains when the lower bound is positive.

Exp.	n	Incumbent $\bar{f}$±SD	Proposal $\bar{f}$±SD	$\bar{\Delta}$	$\mathrm{LCB}_{1-\delta}$	Decision
C012	100	21.41 ± 7.42	20.86 ± 7.62	-0.55	$+0.009$	**Accept**
C014	80	21.38 ± 7.46	20.83 ± 7.66	-0.55	$+0.003$	**Accept**
C012*	80	21.38 ± 7.46	20.83 ± 7.66	-0.55	-0.004	Reject

terministic optimization. It also highlights a practical tradeoff: empirical Bernstein is useful in low-variance settings, whereas Hoeffding can be less conservative under high variance.

5 Discussion

Core findings. Our experiments support three main claims about the Statistical Gödel Machine (SGM): (1) *Statistical safety for recursive edits:* SGM consistently enforces a confidence-based safeguard, rejecting harmful modifications while certifying genuine improvements. This establishes the first practical safety layer for recursive self-modification. (2) *Event-triggered risk allocation improves power:* Confirm-Triggered Harmonic Spending (CTHS) concentrates the error budget on actual confirmation events, enabling early certification of true gains (CIFAR-100) that harmonic schedules miss. (3) *Cross-domain robustness:* SGM generalizes across supervised learning (CIFAR-10/100, ImageNet-100), reinforcement learning (CartPole, LunarLander), and black-box optimization (Rastrigin20), demonstrating consistent risk control and certifying reproducible improvements under diverse conditions.

Why different from prior testing. Standard sequential testing or online FDR governs temporary hypotheses: once a test ends, its mistakes do not persist. In contrast, SGM governs *irreversible self-modifications*: each accepted edit permanently rewrites the incumbent and propagates forward, so guarantees must hold not only per test but cumulatively across a recursive sequence of edits. To further strengthen this contract, we introduce *Confirm-Triggered Harmonic Spending (CTHS)*, which concentrates error budget on the subset of rounds that escalate to confirmation. Unlike classic harmonic splits that spend in every round, CTHS allocates only when an edit is at stake, improving statistical power without exceeding global risk. This combination of irreversibility and event-triggered spending fundamentally distinguishes SGM from prior work.

5.1 Limitations and Future Work

Assumptions. Our guarantees rely on bounded, i.i.d. paired differences and a stable evaluation harness. Real-world pipelines may exhibit heavy-tailed noise, temporal correlation, or drift; in such cases, our certificates remain valid but conservative. Extending SGM with variance-robust or drift-aware bounds is an important next step.

Empirical scope. We validated SGM on small-to-mid scale benchmarks (CartPole, LunarLander, Rastrigin, CIFAR-10/100, ImageNet-100). We did not include very large-scale domains such as ImageNet-1k, Mujoco, or LLM-based agent loops. Thus, the current results demonstrate feasibility rather than ultimate scalability. Applying SGM to these high-stakes pipelines is a natural extension.

Proposer design. Our proposers are deliberately simple (preset or random hyperparameter tweaks), to isolate the gate's guarantees. Stronger proposers—e.g., Bayesian optimizers or learned mutation policies—are fully compatible with SGM and may yield richer dynamics. Studying this interaction is future work.

Compute tradeoffs. Confirmation protocols (e.g., 30 seeds for CIFAR-100) are rigorous but costly. In large-scale settings, adaptive or cost-aware certificates (e.g., α-spending with early stopping) could reduce overhead without compromising safety.

Outlook. SGM establishes the *first statistical safety layer* for recursive self-modification. Its present role is to ground the concept with principled guarantees and cross-domain feasibility. Future work should scale the approach, relax assumptions, and integrate stronger proposers, with the broader goal of making self-improving systems both more capable and reliably safe.

6 Conclusion

We presented the Statistical Gödel Machine (SGM), a practical relaxation of Gödel's vision of self-referential improvement. Whereas classical Gödel machines demand formal logical proofs—unattainable in stochastic, high-dimensional settings—SGM replaces them with statistical confidence certificates, making Gödelian self-reference tractable for modern machine learning pipelines.

Our analysis establishes both per-edit and cumulative guarantees, ensuring that the probability of adopting a harmful modification remains bounded even under indefinite horizons of recursive improvement. A key innovation is *Confirm-Triggered Harmonic Spending (CTHS)*, which preserves familywise error control while concentrating the error budget on confirmation events, thereby improving power on promising edits without exceeding the global budget.

Experiments across reinforcement learning, black-box optimization, and supervised learning—including a 30-seed CIFAR-100 stress test—demonstrate that SGM reliably rejects spurious gains while certifying genuine ones. By combining the conceptual rigor of Gödel machines with the tools of statistical learning, SGM provides a principled, domain-agnostic safety layer for continual self-modification. This is a first step: establishing feasibility and guarantees today, while pointing toward future systems capable of scaling recursive self-improvement to high-stakes real-world applications.

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. PMLR, 2017.

Eitan Altman. *Constrained Markov decision processes*. Routledge, 2021.

Peter Armitage, CK McPherson, and BC Rowe. Repeated significance tests on accumulating data. *Journal of the Royal Statistical Society: Series A (General)*, 132(2):235–244, 1969.

Rina Foygel Barber and Emmanuel J Candès. Controlling the false discovery rate via knockoffs. 2015.

Stefan Falkner, Aaron Klein, and Frank Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. In *International conference on machine learning*, pp. 1437–1446. PMLR, 2018.

William Fithian, Dennis Sun, and Jonathan Taylor. Optimal inference after model selection. *arXiv preprint arXiv:1410.2597*, 2014.

Dean P Foster and Robert A Stine. α-investing: a procedure for sequential control of expected false discoveries. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(2): 429–444, 2008.

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Alexander L Gaunt, Marc Brockschmidt, Rishabh Singh, Nate Kushman, Pushmeet Kohli, Jonathan Taylor, and Daniel Tarlow. Terpret: A probabilistic programming language for program induction. *arXiv preprint arXiv:1608.04428*, 2016.

Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18, 2003.

Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking bbob-2009. In *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*, pp. 1689–1696, 2010.

Nikolaus Hansen, Anne Auger, Raymond Ros, Olaf Mersmann, Tea Tušar, and Dimo Brockhoff. Coco: A platform for comparing continuous optimizers in a black-box setting. *Optimization Methods and Software*, 36(1):114–144, 2021.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

Steven R Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform chernoff bounds via nonnegative supermartingales. 2020.

Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

Ramesh Johari, Pete Koomen, Leonid Pekelis, and David Walsh. Always valid inference: Continuous monitoring of a/b tests. *Operations Research*, 70(3):1806–1821, 2022.

Xavier Leroy. Formal verification of a realistic compiler. *Communications of the ACM*, 52(7): 107–115, 2009.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740*, 2009.

George C Necula. Proof-carrying code. In *Proceedings of the 24th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pp. 106–119, 1997.

Aaditya Ramdas, Tijana Zrnic, Martin Wainwright, and Michael Jordan. Saffron: an adaptive algorithm for online control of the false discovery rate. In *International conference on machine learning*, pp. 4286–4294. PMLR, 2018.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pp. 4780–4789, 2019.

Jürgen Schmidhuber. Gödel machines: Fully self-referential optimal universal self-improvers. In *Artificial general intelligence*, pp. 199–226. Springer, 2007.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2015.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

Ian Waudby-Smith and Aaditya Ramdas. Estimating means of bounded random variables by betting. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 86(1):1–27, 2024.

Roman V Yampolskiy. From seed ai to technological singularity via recursively self-improving software. *arXiv preprint arXiv:1502.06512*, 2015.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

A Appendix

A.1 Hoeffding Bound for Acceptance

We recall the classical Hoeffding inequality.

Theorem 3 (Hoeffding inequality, mean form). *Let $X_1, \ldots, X_n$ be independent with $X_i \in [a, b]$ and mean μ. Let $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} X_i$. Then, for any $\epsilon > 0$,*

$$\Pr\big(\mu \le \hat{\mu} - \epsilon\big) \;\le\; \exp\Big(-2n\epsilon^2/(b-a)^2\Big).$$

Equivalently, with probability at least $1 - \delta$,

$$\mu \;\ge\; \hat{\mu} - (b-a)\sqrt{\tfrac{1}{2n}\ln\tfrac{1}{\delta}}.$$

Specialization to the mean. If all variables share the same bounded range $[a, b]$, then $\sum_{i=1}^{n}(b_i - a_i)^2 = n(b-a)^2$. Letting $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} X_i$ and $\mu = \mathbb{E}[X_i]$, we obtain

$$\Pr(\hat{\mu} - \mu \;\ge\; \epsilon) \;\le\; \exp\Big(-\tfrac{2n\epsilon^2}{(b-a)^2}\Big).$$

Equivalently, with probability at least $1 - \delta$,

$$\mu \;\ge\; \hat{\mu} - (b-a)\sqrt{\tfrac{1}{2n}\ln\tfrac{1}{\delta}}.$$

This one-sided lower-tail bound underlies our acceptance rule in Sec. 3.2.

A.2 CartPole Details

A.3 LunarLander-v2 Details

A.4 CIFAR-10 Details

B Compute Environment

All experiments were run on a standardized cloud image with the following stack:

- PyTorch 2.1.2, Python 3.10 (Ubuntu 22.04), CUDA 11.8
- GPU: RTX 4090D (24GB) × 1, with on-demand scaling
- CPU: 18 vCPU AMD EPYC 9754 128-Core Processor
- Memory: 60GB

This environment was used consistently across all tasks. Only the ImageNet-100 experiments were compute-intensive; all other tasks required modest resources.

Table 7: CartPole-v1: Per-iteration results of PAC-based safeguard. All proposals were rejected, consistent with the strong baseline (493.4 ± 12.6). Hyperparameter changes are shown vs. incumbent.

Iter	Proposal Mean ± SD	$\bar{\Delta}$	$\mathrm{LCB}_{1-\delta}$	Decision	Changes vs. incumbent
1	450.3 ± 42.7	-0.0861	-0.789	Reject	lr: $3\times10^{-4} \rightarrow 1.82\times10^{-3}$; clip: $0.20 \rightarrow 0.35$; nsteps: 2k→ 4k
2	439.8 ± 37.9	-0.1071	-0.860	Reject	lr: $3\times10^{-4} \rightarrow 1.82\times10^{-3}$; clip: $0.20 \rightarrow 0.31$; nsteps: 2k→ 4k
3	455.9 ± 58.7	-0.0749	-0.856	Reject	lr: $3\times10^{-4} \rightarrow 1.75\times10^{-3}$; clip: $0.20 \rightarrow 0.32$; nsteps: 2k→ 4k
4	438.2 ± 52.0	-0.1104	-0.910	Reject	lr: $3\times10^{-4} \rightarrow 1.78\times10^{-3}$; clip: $0.20 \rightarrow 0.35$; nsteps: 2k→ 4k
5	463.6 ± 65.9	-0.0595	-0.874	Reject	lr: $3\times10^{-4} \rightarrow 1.82\times10^{-3}$; clip: $0.20 \rightarrow 0.35$; nsteps: 2k→ 4k
6	439.8 ± 37.9	-0.1071	-0.933	Reject	lr: $3\times10^{-4} \rightarrow 1.82\times10^{-3}$; clip: $0.20 \rightarrow 0.31$; nsteps: 2k→ 4k
7	455.9 ± 58.7	-0.0749	-0.911	Reject	lr: $3\times10^{-4} \rightarrow 1.75\times10^{-3}$; clip: $0.20 \rightarrow 0.32$; nsteps: 2k→ 4k
8	438.2 ± 52.0	-0.1104	-0.954	Reject	lr: $3\times10^{-4} \rightarrow 1.78\times10^{-3}$; clip: $0.20 \rightarrow 0.35$; nsteps: 2k→ 4k

Proposal means and SDs are computed across 19 seeds per iteration. Hyperparameter changes are relative to the incumbent.

Table 8: Accepted proposals for LunarLander-v2. Both runs were warm-started from manually seeded hyperparameters (see note).

Exp.	Iter	$\bar{\Delta}$	$\mathrm{LCB}(1-\delta)$	Mean Reward ± SD	ΔR (mean ± sd [min–max])
B3-ws2	1	0.7108	0.0384	–	$+513.2 \pm 306.9$ [165.1–1344.8]
B5-ws3	1	0.7115	0.0222	–	$+513.8 \pm 306.5$ [165.8–1344.6]

Warm-start origins: B3-ws2 seeded from lr≈ 1.87×10^{-3}, clip≈ 0.343; B5-ws3 seeded from neighborhood around B3-ws2 (lr≈ 1.78–1.86 $\times 10^{-3}$, clip≈ 0.343–0.352). (Common: batch=64, n_steps=4096)

Table 9: CIFAR-10, PAC-EB safeguard). Proposal reduces batch size relative to incumbent. Both frozen prefix ($n = 25$) and full overlap ($n = 30$) yield positive bounds, leading to ACCEPT decisions.

Exp.	n	Inc. Acc. (%) ± SD	Prop. Acc. (%) ± SD	($\bar{\Delta}$)	$\mathrm{LCB}_{1-\delta}\left(\hat{\Delta}\right)$	Decision
BS-64 (N=25)	25	85.5 ± 0.7	87.85 ± 0.18	+2.35	+0.03	ACCEPT
BS-64 (N=30)	30	85.54 ± 0.66	87.86 ± 0.19	+2.32	+0.18	ACCEPT

Algorithm 2 SGM outer loop: proposals certified by fixed-δ or anytime e-value tests for adoption.

```
Require: Init. config $\theta_0$; proposer $\Pi$; harness $\mathcal{H}$; max rounds $T$; risk $\delta \in (0,1)$; proposal period $K$
Ensure: Final config $\theta^\star$; registry $\mathcal{R}$ of accepted/rejected edits
$\theta \leftarrow \theta_0$;   $\mathcal{R} \leftarrow \{(\theta_0, \texttt{baseline})\}$;   $W \leftarrow 1$        ▷ anytime wealth
for $t = 1$ to $T$ do
    $(\mathcal{C}, \mathbf{m}) \leftarrow$ RunInnerLoop$(\theta, \mathcal{H})$        ▷ $\mathcal{C}$: incumbent cache; $\mathbf{m}$: metrics
    if Stagnant$(\mathbf{m})$ or $t \bmod K = 0$ then        ▷ plateau or every $K$ rounds
        $\Theta_{\text{cand}} \leftarrow$ ProposeEdit$(\Pi, \theta, \mathcal{R})$
        for all $\theta' \in$ RankCandidates$(\Theta_{\text{cand}})$ do
            $\{\delta_i\}_{i=1}^{n} \leftarrow$ PairedEvaluate$(\mathcal{H}, \theta, \theta', \mathcal{C})$        ▷ per-seed diffs $\delta_i$ for $n$ trials
            $\bar{\delta} \leftarrow \frac{1}{n}\sum_{i=1}^{n} \delta_i$        ▷ mean improvement
            (LCB, $W$) $\leftarrow$ Certify$(\{\delta_i\}, \alpha, W)$
            if LCB $> 0$ then
                accept and promote
            else
                reject (or continue sampling if enabled)
return $(\theta^\star, \mathcal{R}) \leftarrow (\theta, \mathcal{R})$
```

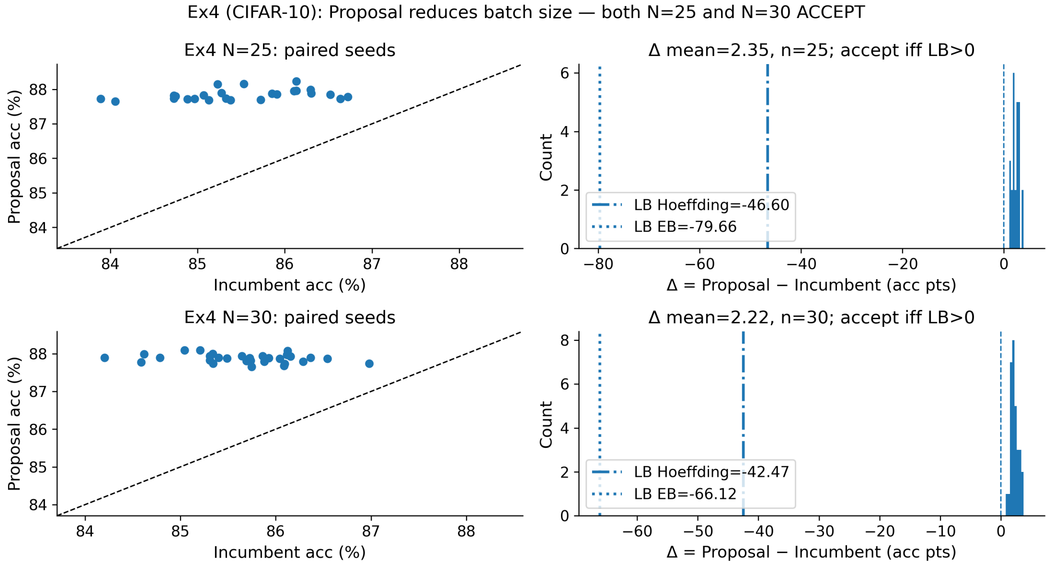

Figure 5: CIFAR-10. Test accuracy vs. epochs for baseline (batch 768) and proposal (batch 64). The proposal yields a small, consistent accuracy gain; acceptance is triggered when $\mathrm{LCB}_{1-\delta}(\hat{\Delta}) > 0$ (both $n = 25$ and $n = 30$).