

Sleep as a Safe Consolidator: Replay Retains, Quarantined Dreaming Covers, Offline Recompute Stabilizes, and Composing All Three Beats Maximizing Any One

Cercle Digital (independent research)

Correspondence: research@cercle.digital

Abstract

A continually learning agent faces a triple bind: online learning on a non-stationary stream *forgets* what it learned earlier; generative self-training *contaminates* its memory with hallucinations whose rate grows with the volume generated; and recomputing its world model while acting is slow and unstable. We show these are three jobs of one offline phase, each with its own lever, and that composing them under a shared budget dominates specializing in any one. Throughout, *replay* means re-training offline on stored past interaction traces; a *quarantine channel* is a store of unverified hypotheses that is never written to the agent’s belief memory; and *do()* denotes verification by execution, running a candidate on chosen inputs and reading its actual output rather than trusting a generator. **(1) Replay retains:** at equal waking interactions and equal offline compute, full replay retains held-out competence at 0.89 ± 0.02 against 0.19 ± 0.01 for online learning, with a monotone dose-response in the replay horizon; the lever is replayed memory, not extra compute. **(2) Quarantined dreaming covers:** high-volume generation written only to the quarantine channel and disposed at dawn by execution yields integrated hallucination 0.000 at every volume while behavioural coverage climbs from 0.10 to 0.94; generative drift becomes a coverage dial precisely because it is quarantined, and prioritized verification is $5.8\times$ more sample-efficient than verifying online at equal budget (0.93 vs 0.16). **(3) Offline recompute stabilizes:** running the same value recomputation offline rather than interleaved with acting reaches criterion $\sim 4\times$ earlier (1.1 vs 4.1 episodes) with $\sim 16\times$ lower variance (0.011 vs 0.177) at equal compute. **(4) The composed phase** (composite 0.84) beats no-sleep (0.38) and every single-job allocation (0.41–0.64): the lifelong score needs all three axes and each job’s returns are concave, so breadth beats depth, without interference. Safety never lives in the generator; it lives in the execution that verifies. Substrates are compact synthetic families that isolate each mechanism; we state that bound explicitly. A self-contained code supplement reproduces every number.

1 Introduction

An agent that learns for its whole life meets its data as a stream, and the stream is *non-stationary*: what it sees today differs from what it saw last week. Three failures follow, and they are usually studied apart. First, learning online on such a stream causes *catastrophic forgetting* (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017): each gradient step on today’s data erodes the weights that encoded yesterday’s. Second, an agent that improves by training on its own generated samples (self-training) risks *contamination*: the more candidates it generates against an under-determined objective, the more confident-but-wrong ones it commits, and the error compounds. Third, when the agent revises its internal model of the world (splitting or merging states, re-propagating value), doing so while it is still acting means acting on a half-updated model, which is both slow to pay off and unstable.

This paper’s claim is that these three failures are answered by one thing an agent does *offline*, between bouts of acting, and that the three answers are distinct jobs of a single scheduled phase we call sleep. The framing is deliberately literal rather than metaphorical: sleep here is not an objec-

tive the agent optimizes, but a *schedule*, a block of cheap offline compute that spends no new interaction with the world. Each job is validated on its own, under controls that hold the scarce resource (interaction) and the compute fixed so that only the job’s own lever varies; then the three are composed under a shared budget to ask whether breadth or depth wins. The research question in one sentence: *are forgetting, self-training contamination, and unstable recompute three separable jobs of one offline consolidation phase, and does running all three under a shared budget beat maximizing any one?*

Terminology. Several terms recur from the title onward. A *continual* (or lifelong) learning agent trains on a stream of tasks without revisiting them at will (Parisi et al., 2019; De Lange et al., 2021). *Replay* (Lin, 1992; Robins, 1995) means re-training offline on stored traces of past interactions, interleaving old data with new so the weights are not overwritten; it is the algorithmic counterpart of the memory-consolidation role attributed to non-REM sleep (McClelland et al., 1995; Diekelmann & Born, 2010). A *generator* here is any model that proposes candidate solutions (programs, plans); *dreaming* is running it at high volume offline, in the

sense of the wake-sleep family (Hinton et al., 1995; Ellis et al., 2021). A *quarantine channel* is a store of *unverified* candidates kept strictly separate from the agent’s belief memory: nothing enters belief until it is checked. `do()` denotes a Pearl-style intervention (Pearl, 2009): for a candidate program, executing it on chosen inputs and reading its actual output, as opposed to trusting the generator’s confidence. *Offline recompute* is running the agent’s planning update (here value iteration) between episodes rather than interleaved with acting. Finally, a *compute-matched, interaction-matched* control gives every arm the same number of gradient steps and the same number of fresh interactions, so a difference between arms is attributable only to the pre-registered lever, never to spending more.

Contributions. Measured, in order (Figure 1):

1. **Replay retains** (Section 4). On a non-stationary stream, at equal interactions and equal compute, full nightly replay holds cumulative held-out accuracy at 0.89 ± 0.02 where online learning collapses to 0.19 ± 0.01 (retaining only the most recent family), with a monotone dose-response in the replay horizon: the lever is replayed memory, not extra compute (bench NREM-REPLAY).
2. **Quarantined dreaming covers** (Section 5). Committing raw dreams hallucinates increasingly with volume ($0.035 \rightarrow 0.155$); writing them only to the quarantine channel and verifying at dawn by `do()` keeps committed hallucination at 0.000 at every volume while coverage climbs $0.10 \rightarrow 0.94$; verifying a ranked pool is $5.8\times$ more efficient than verifying online at equal budget (bench REM-QUARANTINE).
3. **Offline recompute stabilizes** (Section 6). At equal value-iteration budget, recomputing the map offline reaches criterion $\sim 4\times$ earlier (1.1 vs 4.1 episodes) with $\sim 16\times$ lower variance (0.011 vs 0.177) than interleaving it with acting; the lever is the schedule, not the recomputation (bench SURGICAL-NIGHT).
4. **Composition beats specialization** (Section 7). Under a shared night budget, running all three jobs (composite 0.84) beats no-sleep (0.38) and every single-job allocation (0.41–0.64); the lifelong score needs all axes and each job is concave, so breadth beats depth, without interference (bench SLEEP-ORGAN).

Scope claim. Every result is measured on compact synthetic substrates chosen to *isolate* one mechanism each: an operation-recognition stream for forgetting, a small program-synthesis language for dreaming, a controlled prerequisite chain for recompute, and a composition harness driven by the measured single-job curves for the budget question. We claim the mechanisms and their composition at this grain, with every number reproducible; we do not claim a single

heavy end-to-end agent, and Section 9 states what that would take.

2 Related work

Complementary learning systems. The cognitive anchor is the complementary-learning-systems account (McClelland et al., 1995; Kumaran et al., 2016): a fast hippocampal store of specific episodes and a slow neocortical learner of structure, with interleaved replay from the former consolidating the latter and thereby avoiding catastrophic interference. Our replay job is that account made an explicit schedule and measured against a compute-matched online control; the contribution is not the idea of replay but the isolation of *replayed memory*, as opposed to extra compute, as the lever.

Continual learning and replay. Catastrophic forgetting (McCloskey & Cohen, 1989) is countered by three method families (Parisi et al., 2019; De Lange et al., 2021): regularization (elastic weight consolidation, Kirkpatrick et al., 2017), replay (pseudorehearsal, Robins, 1995; experience replay, Lin, 1992; Mnih et al., 2015; prioritized replay, Schaul et al., 2016), and parameter isolation. Generative replay (Shin et al., 2017; van de Ven et al., 2020) trains a generator to synthesize past-task samples and replays those. The distinction that organizes this paper is that generative replay’s samples are learned *without verification*: the generator’s mistakes enter the solver’s training set. Our dreaming job separates the two roles: the generator proposes into quarantine, and `do()` disposes at dawn, so generation buys coverage while execution owns safety.

Wake-sleep and offline recompute. The wake-sleep algorithm (Hinton et al., 1995) alternates a waking recognition phase with a sleeping generative phase; DreamCoder (Ellis et al., 2021) makes the sleep phase build a library of abstractions for program synthesis, a close cousin of our recompute job (offline restructuring of the agent’s own representation). Offline recomputation of a value function from a learned model is Dyna (Sutton, 1991); standard value iteration is textbook (Sutton & Barto, 2018). What we add is the *timing* claim measured against an interleaved control at equal compute: the same recomputation is faster and far more stable when batched offline.

Self-training safety. Pseudo-labeling methods gate self-generated labels by a confidence threshold (Sohn et al., 2020); the safety is a property of the model’s own confidence. Our quarantine gates by *external execution* rather than internal confidence, which matters because a generator’s confidence is not a reliable proxy for behavioural correctness on an under-determined specification. This is the

same wager as the rest of the interventional-agent program: verification is an intervention, not a belief.

Sleep neuroscience. The biological literature attributes memory consolidation to sleep (Walker & Stickgold, 2004; Diekelmann & Born, 2010), and the overfitted-brain hypothesis (Hoel, 2021) proposes that dreaming combats overfitting by injecting corrupted, out-of-distribution inputs. Our results are an engineering echo, not a biological claim: we measure that an offline replay-plus-generate plus-recompute schedule pays on artificial agents, and we take the three-way functional decomposition (retain, cover, stabilize) as the useful structure, without asserting it mirrors sleep stages.

3 Substrates and metrology

The four substrates. Each bench isolates one mechanism on the smallest faithful terrain. NREM-REPLAY uses an *operation recogniser*: given the input and output of an element-wise integer operation (one of eleven: multiply, add or subtract by a small constant, square, negate) applied to a random list, a small multilayer network predicts which operation produced it; labels come from executing the operation, so they are causal ground truth. REM-QUARANTINE uses a small *program-synthesis language*: programs are short chains of modular integer operations; a goal shows a weak (under-determined) input-output specification plus held-out inputs that discriminate look-alikes; behaviour is decided by execution. SURGICAL-NIGHT uses a controlled *prerequisite chain*: rooms 0 to R with reward only at R , where a synchronous value-iteration sweep propagates the reward one room backward, so convergence takes about R sweeps. SLEEP-ORGAN is a *composition harness*: three compact simulators, each faithful to the measured single-job curve (real forgetting-and-replay dynamics, the measured quarantine verified-hit probability, the measured online-to-offline stability endpoints), driven by an allocated share of a shared night budget.

Metrology. Every bench follows the same discipline: fixed seeds with explicit random-number generators; 10 to 16 seeds with 95% confidence intervals reported as mean $\pm$ half-width; *compute-matched and interaction-matched controls by construction* (arms differ only in the pre-registered lever, so a monotone effect isolates that lever and rules out “more compute” as a confound); pre-registered hypotheses with GO/NO-GO thresholds fixed before the run and printed by each script; strong controls rather than strawmen (an online arm at equal budget, a raw-commit arm, a random-skip arm, a never-recompute floor); and accepted nulls: a hypothesis that fails is reported with its diagnosed mechanism. Gains here are gains of *efficiency and stability*, never of safety, which is owned by execution; we state this

throughout. Bench names are the script names of the code supplement (Appendix A).

4 Replay retains

The first job is consolidation: does replaying stored traces offline prevent forgetting, and is the lever the replayed memory rather than the extra compute a night affords?

Hypothesis (NL). (H1) On a non-stationary stream, cumulative held-out accuracy over all families seen is CI-above the online baseline for full replay, at equal interactions *and* equal compute. (H2) The effect is a dose-response in the replay horizon (how far back the night reaches): accuracy is monotone non-decreasing in the horizon, isolating replayed memory as the lever. (H3, reported) the forgetting signature: online retains only the most recent family; full replay holds all families.

Experiment (operationalisation). Non-stationary curriculum: four days, one new operation family per day (eleven operations, chance 0.091). Every arm acquires the same $B = 60$ fresh labelled interactions per day (equal interactions) and takes the same 300 gradient steps per day (equal compute); arms differ only in the *replay horizon*, the set of past days the night’s gradient steps may sample: horizon 0 samples today only (online), horizon 1 today plus the previous day, horizon ALL every stored day (full replay). Metric: held-out accuracy on fresh inputs over all families, plus per-family retention. 10 seeds. (Bench NREM-REPLAY.)

Result. Figure 2. Online learning reaches cumulative accuracy 0.193 ± 0.011 and, in the per-family breakdown, remembers only the last day’s family (retention $[0.00, 0.00, 0.00, 1.00]$ from oldest to newest): the textbook catastrophic-forgetting signature. A one-day replay horizon already recovers the two most recent families (0.420 ± 0.027), and full replay reaches **0.891 ± 0.018** with balanced retention $[0.84, 0.92, 0.91, 0.90]$. The dose-response is monotone ($0.19 \rightarrow 0.42 \rightarrow 0.89$) at fixed compute, so the gain is attributable to replaying older memory, not to spending more gradient steps.

Conclusion. Replay is the consolidation job of the offline phase, and its lever is the memory horizon. It is *free in interactions*, the scarce resource: it re-uses cheap offline compute on already-acquired traces and adds no capability a full-replay online learner would not have; what it changes is that the day’s interactions, replayed at night, keep paying instead of being overwritten. The moment the stream is non-stationary (which for a lifelong agent is always), consolidation must run by default.

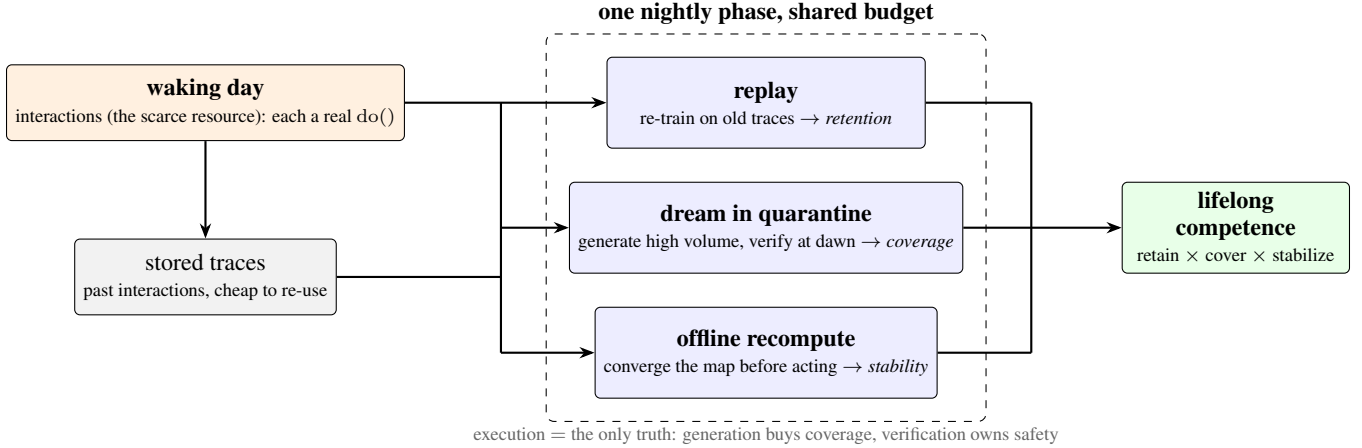


Figure 1: One offline phase, three jobs, one shared budget. A waking day spends the scarce resource (fresh interactions with the world, each a real `do()`) and stores the traces. At night, one scheduled phase runs three offline jobs that spend no new interaction: *replay* re-trains on stored traces and retains past competence; *quarantined dreaming* generates candidates at high volume into a store that never touches belief memory, and verifies the best by execution at dawn, buying behavioural coverage; *offline recompute* re-converges the agent’s planning map before it acts, buying stability. Under a fixed shared budget, running all three beats pouring the budget into any one (Section 7).

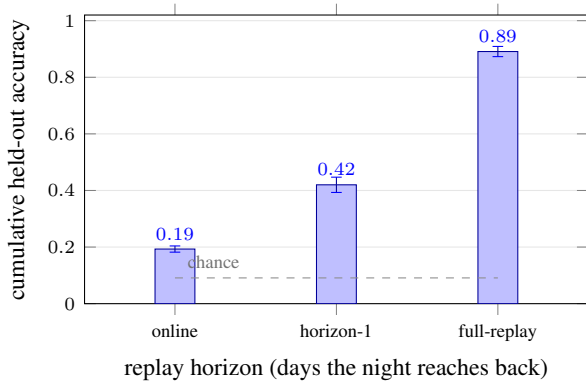


Figure 2: Replay dose-response (bench NREM-REPLAY; 10 seeds, 95% CI; non-stationary stream of four operation families, chance 0.091). At equal interactions and equal compute, cumulative held-out accuracy rises monotonically with how far back the night replays; only the replay pool differs between bars. Online (horizon 0) forgets all but the newest family; full replay retains all four.

5 Quarantined dreaming covers

The second job is generation. A generator that improves the agent by proposing candidates is useful only if its inevitable mistakes never reach belief. The wake-sleep split makes that separation architectural: dream at night into a quarantine channel, verify at dawn by execution.

Hypothesis (NL). (H1, safety) Committing raw dreams hallucinates, and worsens as the generated volume grows; writing dreams only to quarantine and verifying at dawn keeps committed hallucination at 0.000 at every volume. (H2, coverage) Quarantine solve rate rises with the vol-

ume dreamed. (H3, economy, reported) At equal dawn-verification budget, verifying the top of a large ranked dream pool beats verifying the first candidates online.

Experiment (operationalisation). Program-synthesis language: three-operation chains modulo five; each goal shows one weak input-output pair (under-determined, so behavioural look-alikes exist) plus five held-out inputs that discriminate. For each nightly volume $N \in \{4, 16, 64, 256\}$: the *raw* arm commits a candidate that matches the weak specification without further checking; the *quarantine* arm verifies each weak-matcher on the held-out inputs by execution (`do()`) and commits only a verified one, so it never commits an unverified candidate by construction. Hallucination is the fraction of commits that fail on held-out inputs; coverage is the fraction of goals solved with a verified program. Economy arm: at a fixed dawn budget of $B = 8$ verifications, online verifies the first 8 candidates as they arrive, while sleep ranks a pool and verifies the top 8. 80 goals per seed, 10 seeds. (Bench REM-QUARANTINE.)

Result. Figure 3. Raw commit hallucination climbs with volume, from 0.035 ± 0.017 at $N=4$ to 0.155 ± 0.023 at $N=256$: sampling more against a weak specification finds more confident look-alikes that are wrong on held-out inputs. Quarantine hallucination is 0.000 at every volume, by construction, since nothing unverified is committed. Meanwhile coverage rises with volume, from 0.10 ± 0.02 to 0.94 ± 0.01 : dreaming more finds more goals that admit a verified solution. The economy arm confirms the efficiency: at equal dawn-verification budget, sleep verifying the top of a ranked pool solves 0.93 ± 0.01 against 0.16 ± 0.03 for online verification of the first candidates, a $5.8\times$ gain.

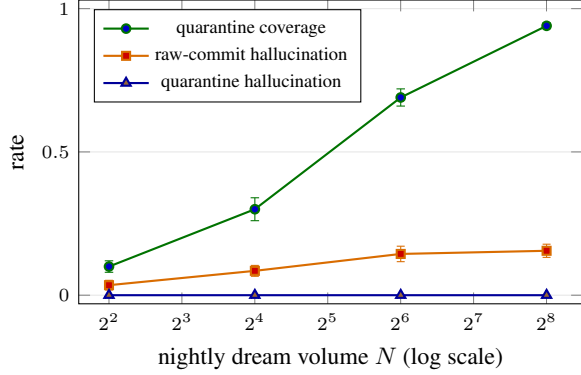


Figure 3: Dreaming buys coverage, quarantine keeps it safe (bench REM-QUARANTINE; 10 seeds, 95% CI). As nightly volume grows, committing raw dreams hallucinates increasingly (orange), while quarantine plus dawn verification stays at zero hallucination (blue) and its coverage climbs (green). Generative drift is a coverage dial only because it is quarantined.

Conclusion. Safety is invariant to the volume dreamed because it never lived in the generator; it lives in the dawn’s execution. Coverage, by contrast, is a dial the generator turns, precisely because its output is quarantined until verified. The rule “generate more only helps if you verify” becomes an architecture: dream freely at night, commit only what execution certifies at dawn. A learned rather than uniform generator would sharpen the ranking (the economy), not the safety, which execution owns regardless.

6 Offline recompute stabilizes

The third job is planning. When the agent revises its map (splitting or merging states, then re-propagating value), the value function must re-converge. Doing that while acting means acting on a map that has not finished updating. This bench isolates the *timing*: the same recomputation and the same total compute, scheduled offline versus interleaved.

Hypothesis (NL). (H1) Offline recompute reaches criterion sooner: time-to-criterion (episodes to first solve) is CI-below the interleaved arm, at equal sweep budget. (H2) Offline recompute is more stable: per-episode reach variance is CI-below the interleaved arm. (H3, sanity) both beat the never-recompute floor, and offline is at least as good as interleaved on late reach.

Experiment (operationalisation). Prerequisite chain of depth $R=8$, reward only at the last room; a synchronous value-iteration sweep propagates the reward one room backward, so about R sweeps are needed to converge. The scarce resource is the sweep budget $S=8$ (compute), equal across arms; the arms differ only in *when* the sweeps are spent.

Table 1: Offline recompute versus interleaved (bench SURGICAL-NIGHT; depth $R=8$, equal sweep budget $S=8$; 16 seeds, 95% CI). Time-to-criterion is episodes to first solve (lower is better); reach variance measures instability (lower is better). Same recomputation, same compute; only the schedule differs.

schedule	time-to-crit.	late reach	reach var.
never (floor)	12.9 ± 2.6	0.04 ± 0.06	0.032 ± 0.017
online (interleaved)	4.1 ± 0.6	1.00 ± 0.00	0.177 ± 0.013
offline (batched)	1.1 ± 0.1	1.00 ± 0.00	0.011 ± 0.012

never: no sweeps, the deep reward never propagates (the floor). *online*: the sweeps are interleaved, one before each acting episode, so the agent acts on a still-propagating map. *offline*: all sweeps are spent in one night before acting, so the agent always acts on a converged map. Metrics: time-to-criterion, late reach (last three episodes), and per-run reach variance. 16 seeds. (Bench SURGICAL-NIGHT.)

Result. Table 1. The never arm confirms the reward does not propagate on its own (late reach 0.04 ± 0.06); both recomputing arms reach the goal (late reach 1.00), so the recomputation matters and only its timing is in question. Offline reaches criterion in 1.1 ± 0.1 episodes against 4.1 ± 0.6 for online ($3.7\times$ sooner), and its per-episode reach variance is **0.011 ± 0.012** against 0.177 ± 0.013 ($16\times$ smaller). The interleaved arm churns because it acts on a map whose deep value has not yet refluxed to where the agent stands; the offline arm converges first, then walks a finished map.

Conclusion. The planning recomputation belongs to the offline phase. At equal compute, batching it offline solves sooner and far more stably than interleaving it with acting: one does not redraw the map while walking on it. This is the job that the individual retention and coverage jobs do not supply, and it is again free in interactions.

7 Composition beats specialization

The three jobs each pay alone. The capstone question is composition: under a single shared night budget, does running all three beat pouring the budget into one, and do the jobs interfere?

Why breadth should win. The lifelong score aggregates the three axes multiplicatively: a life that forgets, or hallucinates, or acts on an unconverged map scores poorly no matter how well it does on the other two. With a geometric-mean aggregate and concave per-axis returns, spreading a fixed budget dominates concentrating it.

Proposition 1 (breadth beats depth). *Let a life score be the geometric mean $g(x_1, x_2, x_3) = (x_1 x_2 x_3)^{1/3}$ of three axis*

scores, each $x_i = f_i(b_i)$ a concave increasing function of the budget b_i allocated to axis i , with a low common baseline $x_i(0) = x_0$. Under a fixed total budget $\sum_i b_i = B$, the even split $b_i = B/3$ yields a strictly higher g than any single-axis allocation $b_j = B$, $b_{i \neq j} = 0$ whenever the concave gain $f_i(B/3) - x_0$ is a sufficiently large fraction of $f_i(B) - x_0$. Concavity guarantees $f_i(B/3) - x_0$ captures most of $f_i(B) - x_0$; the geometric mean punishes the two zero-gain axes of the single allocation, which stay at x_0 .

This is a statement about the aggregate and the concavity, not a claim that the jobs are magically synergistic; they are orthogonal by construction, and the point is exactly that orthogonal concave axes under a product score reward breadth. The bench measures whether the numbers bear it out.

Hypothesis (NL). (H1) The composed phase beats no-sleep: the all-three composite is CI-above sleep-off, at equal waking interactions. (H2) Breadth beats depth: the all-three composite is CI-above the best single-job allocation. (H3, reported) no interference: each single-job allocation maxes its own axis, and the all-three phase keeps every axis above the online baseline.

Experiment (operationalisation). A lifelong agent lives six days with a fixed shared night budget B ; waking interactions are equal across all configurations (sleep is free in interactions). Three jobs, each a compact simulator faithful to its measured single-job curve: retention (replay refreshes the most-decayed families), coverage (concave verified-hit probability in the dreamed volume), stability (1 minus the reach variance interpolated between the measured online and offline endpoints of Section 6). Configurations: sleep-off (all online baselines), each single job given the whole budget, and all-three given $B/3$ each. Score: geometric mean of the three axes. 16 seeds. (Bench SLEEP-ORGAN.)

Result. Figure 4 and Table 2. Sleep-off scores composite 0.38: it forgets (retention 0.32) and undercovers (coverage 0.21). Each single job maxes its own axis and leaves the others at baseline, so its composite is capped: replay-only 0.55 (retention 0.97), dream-only 0.64 (coverage 1.00), recompute-only 0.41 (stability 0.99). The all-three phase reaches composite **0.84**, above no-sleep (0.38) and above the best single job (0.64), while keeping every axis above its online baseline (retention 0.73 > 0.32, coverage 0.89 > 0.21, stability 0.90 > 0.82). The jobs do not interfere: they touch disjoint state (consolidate weights, generate into quarantine, re-converge the map), so sharing the night composes cleanly. (Confidence half-widths are below 0.01 on the composite across 16 seeds.)

Conclusion. The offline phase is a multi-function organ, not a single job. Under a shared budget, its value is run-

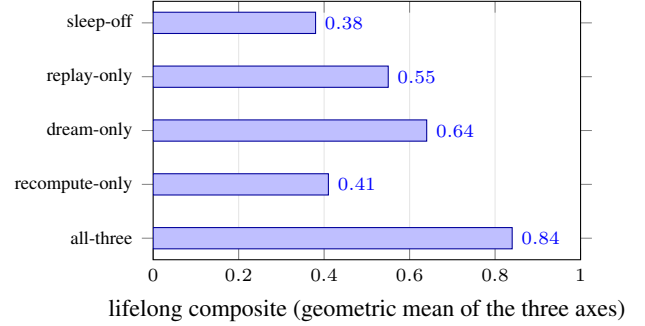


Figure 4: Breadth beats depth (bench SLEEP-ORGAN; 16 seeds; composite CI half-width < 0.01). Under a shared night budget, running all three jobs (top) beats no-sleep and every single-job allocation: the product score needs all axes, and each job is concave, so a third of the budget on each axis dominates the whole budget on one.

Table 2: Axis breakdown of the composition (bench SLEEP-ORGAN; 16 seeds). Each single-job allocation maxes its own axis and leaves the others at their online baseline; the composed phase keeps every axis well above baseline, which the product score rewards.

config	retention	coverage	stability	composite
sleep-off	0.32	0.21	0.82	0.38
replay-only	0.97	0.21	0.82	0.55
dream-only	0.32	1.00	0.82	0.64
recompute-only	0.32	0.21	0.99	0.41
all-three	0.73	0.89	0.90	0.84

ning consolidation, generation, and recompute *together* every night, because the lifelong score needs all three axes and each job’s concave returns make a third of the budget capture most of its gain. Breadth beats depth without interference.

8 Discussion

Why safety lives in verification, not the generator. Every job that could contaminate belief (generative dreaming) is gated by execution, and every job that cannot (replay of already-verified traces, recompute of a value field) is safe by construction. The quarantine channel is the architectural expression of that separation: a candidate is a hypothesis until an intervention makes it a belief. This is why coverage can be driven arbitrarily high by generating more, at zero cost in committed hallucination: the generator is allowed to be wrong because it is never trusted, only verified. Gating by external execution rather than by the generator’s own confidence (Sohn et al., 2020) matters because confidence on an under-determined specification is not a proxy for correctness (Section 5).

The offline phase is a scheduler, not an objective. None of the three jobs adds a capability the agent lacks while awake; each re-spends cheap offline compute on work that would otherwise compete with acting. Replay re-uses stored traces, dreaming re-uses the generator, recompute re-uses the model. The phase’s value is entirely in *when* the work is done: off the critical path of interaction, where compute is cheap and the map can converge undisturbed. This is the same lesson as Dyna’s offline planning (Sutton, 1991) and the wake-sleep split (Hinton et al., 1995), measured here as three distinct wins under matched budgets.

Announced refinements. Two refinements are natural and left for future measurement. Replay here is uniform over history; prioritizing by surprise or by contested evidence (Schaul et al., 2016) should improve the retention economy by replaying the traces that most need it. And the generator is uniform; a learned generator would sharpen the dawn ranking, improving the coverage economy without touching the safety, which execution owns regardless.

9 Limitations

- **Compact substrates, not one heavy agent.** Each bench isolates one mechanism on the smallest faithful terrain (an operation recogniser, a small synthesis language, a controlled chain). These demonstrate the mechanisms and their composition, not a single lifelong agent carrying all three on a shared heavy substrate; porting each job to a common grounded stream is the direct next step.
- **The composition harness is a model.** Section 7 drives three compact simulators, each fit to a measured single-job curve, from a shared budget. Its claim is breadth-versus-depth under concavity and a product score, grounded in the measured curves; it is not an independent end-to-end measurement of three jobs interfering on one substrate, and the axes are orthogonal by construction rather than by observation.
- **Uniform replay and uniform generator.** Replay is uniform over history and the generator is uniform; both are the conservative choice, and both would be sharpened by the announced refinements. The reported economies are therefore lower bounds on what prioritized replay and a learned generator would yield.
- **Fixed class and probe structure.** The recogniser’s operation set is fixed (no admission of new operation types), the synthesis language is small, and behavioural equivalence is decided on a finite probe battery, so verified coverage is coverage on that battery. Admitting new classes and richer probes is orthogonal work.
- **The composition verdict depends on the score.** “Breadth beats depth” follows from a multi-axis

geometric-mean life score; a different aggregator (for instance one that tolerates a zero on one axis) would rank the configurations differently. We state the aggregator explicitly (Proposition 1) so the dependence is visible.

- **Efficiency and stability, not safety.** Every gain reported is an efficiency or stability gain; none is a safety gain, because safety is owned by the waking and dawn interactions (the `do()`). An agent whose interactions are themselves unreliable inherits that unreliability; the offline phase cannot manufacture correctness the interventions do not supply.

10 Conclusion

An agent’s sleep is not a metaphor but a measurable schedule: three offline recomputations that cost no waking interaction, each answering a distinct failure of continual learning, whose composition under a shared budget dominates any specialization. What is demonstrated, each number reproducible from the code supplement: at equal interactions and equal compute, full replay retains held-out competence at 0.89 ± 0.02 where online learning collapses to 0.19 ± 0.01 , monotonically in the replay horizon; high-volume generation written only to a quarantine channel and verified at dawn by execution holds committed hallucination at 0.000 at every volume while coverage climbs $0.10 \rightarrow 0.94$, with prioritized verification $5.8\times$ more efficient than online at equal budget; recomputing the map offline reaches criterion $\sim 4\times$ earlier and $\sim 16\times$ more stably than interleaving it at equal compute; and under a shared night budget, running all three (0.84) beats no-sleep (0.38) and every single-job allocation (0.41–0.64), because the lifelong score needs all axes and each job is concave.

The caveats are part of the result: the substrates are compact and isolate one mechanism each, the composition is measured through a harness fit to the single-job curves, replay and the generator are uniform, and every gain is efficiency or stability rather than safety. What remains open is what the mechanisms most invite. Does the three-way decomposition survive on one heavy grounded substrate that carries retention, coverage, and stability at once, with the jobs genuinely competing for the night? Do prioritized replay and a learned generator turn the reported lower-bound economies into larger ones? And does the product-score verdict, that breadth beats depth, hold under aggregators that weigh the axes differently? Within the measured perimeter, the division of labour stands: the coverage may come from the generator, the stability from the timing, the retention from the replay, but the truth comes only from the verification.

Reproducibility statement

A self-contained code supplement (the `code/` directory accompanying this paper) reproduces every number. It contains a shared `metrology.py` (mean with 95% confidence interval, full seeding, CI comparison) and one `bench_*.py` script per measured section, each printing its pre-registered hypotheses, its controls, and its GO/NO-GO verdict. All substrates are synthetic and regenerated bit-for-bit from (configuration, seed); dependencies are public (numpy, and torch for NREM-REPLAY only); there is no dataset download and no language model anywhere. All random-number generators are explicit. Appendix A maps every bench name in the text to its script.

References

- De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. A continual learning survey: defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2021.
- Diekelmann, S. and Born, J. The memory function of sleep. *Nature Reviews Neuroscience*, 11(2):114–126, 2010.
- Ellis, K., Wong, C., Nye, M., Sablé-Meyer, M., Morales, L., Hewitt, L., Cary, L., Solar-Lezama, A., and Tenenbaum, J. B. DreamCoder: bootstrapping inductive program synthesis with wake-sleep library learning. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation (PLDI)*, pp. 835–850, 2021.
- Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. The “wake-sleep” algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.
- Hoel, E. The overfitted brain: dreams evolved to assist generalization. *Patterns*, 2(5):100244, 2021.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.
- Kumaran, D., Hassabis, D., and McClelland, J. L. What learning systems do intelligent agents need? Complementary learning systems theory updated. *Trends in Cognitive Sciences*, 20(7):512–534, 2016.
- Lin, L.-J. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321, 1992.
- McClelland, J. L., McNaughton, B. L., and O’Reilly, R. C. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419–457, 1995.
- McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: the sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.
- Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: a review. *Neural Networks*, 113:54–71, 2019.
- Pearl, J. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.
- Robins, A. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.
- Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. In *International Conference on Learning Representations (ICLR)*, 2016.
- Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.
- Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. Fix-Match: simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 2020.
- Sutton, R. S. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, 1991.
- Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.
- van de Ven, G. M., Siegelmann, H. T., and Tolias, A. S. Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications*, 11:4069, 2020.
- Walker, M. P. and Stickgold, R. Sleep-dependent learning and memory consolidation. *Neuron*, 44(1):121–133, 2004.

A Bench index

Table 3: Bench names used in the text, their role, their substrate, and the reproducing script in the code supplement.

Bench	Role in this paper	Substrate	Script
NREM-REPLAY	replay retains; dose-response in the horizon	operation-recogniser stream	<code>bench_nrem_replay.py</code>
REM-QUARANTINE	quarantined dreaming covers; safety at dawn	program-synthesis language	<code>bench_rem_quarantine.py</code>
SURGICAL-NIGHT	offline recompute stabilizes; timing lever	controlled prerequisite chain	<code>bench_surgical_night.py</code>
SLEEP-ORGAN	composition beats specialization	composition harness	<code>bench_sleep_organ.py</code>

B The shared-budget composition model

The composition harness of Section 7 realizes each axis as a compact function of its allocated budget share, fit to the measured single-job endpoints.

- **Retention.** A six-day non-stationary stream; each day’s family is learned to near one and then decays by a fixed factor per subsequent day (catastrophic forgetting). The night’s budget sets how many of the most-decayed families are refreshed by replay. Zero budget refreshes nothing (only the newest family stays high); full budget refreshes all, reproducing the online-versus-replay endpoints of Section 4.
- **Coverage.** The verified-hit probability per dreamed candidate is small and fixed; the budget sets the number of dreams N , so coverage is $1 - (1 - p)^N$, a concave increasing function that reproduces the rising, saturating coverage curve of Section 5.
- **Stability.** The reach variance interpolates, concavely in the sweep budget, between the measured interleaved endpoint (0.177) at zero offline budget and the measured offline endpoint (0.011) at full budget (Section 6); the stability score is one minus that variance.

The life score is the geometric mean of the three axes, so a low score on any one axis dominates the product (Proposition 1). Configurations differ only in how the shared budget is split; waking interactions are held equal across all configurations, since the offline phase spends none. The harness therefore measures the budget-allocation question (breadth versus depth under concavity and a product score), taking the per-axis shapes from the three single-job benches rather than re-deriving them.