Exploration Cost: The Exploration Coefficient in Onboard Reinforcement-Learning Schedulers Is an Independent Driver of Battery Aging

John Goodman (OceanSparx)

October 2026

Abstract

Reinforcement-learning (RL) schedulers are now flying on real spacecraft: NASA's Carruthers Geocorona Observatory (launched 24 September 2025, per NASA's own mission status page — not itself a LEO mission; it orbits near the Sun–Earth L1 point) uses deep RL as its default operational scheduler for long-horizon operations under power, thermal, and instrument constraints. We cite it only as evidence that RL scheduling is flight-proven in general, not as a LEO example. For low-Earth-orbit (LEO) platforms specifically, the battery is frequently the lifetime-limiting consumable, and an RL scheduler's exploration-exploitation setting — how often it tries a novel action instead of its current best estimate — is usually tuned purely as a *learning-speed* knob. We show that the exploration coefficient is also an **independent driver of physical wear**, separate from and in addition to its effect on how quickly mission value is delivered. Holding delivered mission value fixed by construction (every scheduler setting runs until it reaches the same cumulative reward, not a fixed wall-clock duration), full exploration costs **1.47–1.71× the wear or aging** of a purely exploitative scheduler, depending on the physical model — battery depth-of-discharge cycling (two models: a power-law damage law and an independently-published log-linear cycle-life law) or, in a third model, mechanical actuator wear on a documented robot platform (Archard's law). A zero-initialised greedy baseline is unfairly handicapped: it locks onto an arbitrary first action and, without a brief warm start, reverses the effect in 13–19 of 30 independent task draws (median ratios 0.73–1.04×, none significant). Every arm here therefore uses a warm start, applied identically at every exploration rate, and results are reported over 30 independent environments per model. With it, the battery result is fully robust (0/30 reversals in both models, Wilcoxon $p = 9.3\times10^{-10}$ each); the actuator-wear generalization is real but less uniform (2/30 reversals, ratio range 0.83–3.30×). We report both the effect and its exception rate, rather than the effect alone. What transfers across all three models is the qualitative claim: exploration costs physical wear independently of task throughput, through both more total actions and (usually, to a lesser degree) higher wear per action. We frame this as a design implication for anyone tuning exploration schedules on hardware with a finite-cycle-life component, not as a validated property of any specific deployed system — and as a demonstration that a single-seed simulation result, however clean it looks, is not evidence until it survives being replayed on environments it was not tuned on.

1. Introduction

Deploying reinforcement learning on physical hardware raises a concern the RL literature treats mostly as a *safety* problem: random exploratory actions risk damaging the hardware they act on, and sim-to-real RL platforms are designed around the expectation that hardware will need frequent repair as a cost of real-world exploration [ROBOTICS-WEAR]. That framing — exploration as a hazard to be tolerated or bounded — is different from asking a quantitative question: *given* that exploration is otherwise safe, does the exploration *rate* itself still cost something measurable, against the same unit of useful work done?

For a satellite, RL scheduling is now flight-proven ([CARRUTHERS]; Section 2), and the natural unit of "cost" for a LEO platform specifically is not actuator wear but battery life. LEO satellites cycle their battery once per orbit (charging in sunlight, discharging through eclipse), dozens of times per day, for a mission that may run years. Battery cycle life is finite and governed by depth of discharge (DoD) per cycle: deeper discharges damage the

battery disproportionately (a power-law or exponential relationship, depending on chemistry and source — see Section 3.2). An onboard RL scheduler deciding which task to run each orbit therefore makes a decision that is, among other things, a decision about how deep that cycle's discharge will be.

The motivation is practical. Efficiency gains in applied engineering often come from noticing that two quantities normally tuned separately are coupled, rather than from improving either alone. The exploration coefficient is conventionally tuned for one objective, learning speed, yet it may also be coupled to a second, hardware wear, that a learning-speed-only tuning cannot see. If that coupling is real, the two can be tuned jointly, which is a design option that is otherwise invisible.

The question this paper asks: **does the exploration-exploitation setting of such a scheduler drive battery aging independently of how much mission value it delivers?** If exploration simply trades "doing more work" for "aging the battery faster," that is not a finding — doing more work costs more energy by definition. The interesting claim, and the one this paper tests, is narrower: that at a **fixed delivered mission value**, a more exploratory scheduler still costs more battery life, through mechanisms (more total cycles needed, deeper cycles on average) that a purely exploitative scheduler does not pay.

2. Related Work

Exploration-induced hardware wear (robotics). The RL-for-robotics literature acknowledges that exploratory actions wear physical components and generally addresses this through constrained or safe exploration, or by over-provisioning hardware, rather than by measuring the wear cost against delivered task value [ROBOTICS-WEAR]. Generalized state-dependent exploration (gSDE), for example, was introduced to reduce the jerky, damaging motion that unstructured exploration produces on real robots, and trades smoothness against performance through a noise resampling interval [SMOOTH-EXPLORATION]; it does not quantify wear against delivered reward. We are not aware of prior work that holds delivered reward fixed and decomposes the resulting wear/aging into independent mechanisms, which is the contribution here, applied to a different hardware failure mode (battery cycling rather than mechanical wear).

Battery degradation in RL reward functions. A substantial literature on RL for electric vehicle charging incorporates battery degradation directly into the *reward*. [EV-CHARGING-RL], for instance, uses soft actor-critic (SAC) to plan second-life-battery charging-station operation under uncertain arrivals and prices; SAC's entropy-regularized objective balances exploration against exploitation as part of how it optimizes that reward, though the cited paper's own framing is about the planning problem, not about exploration-exploitation tuning specifically. The broader pattern — making the agent reward-aware of degradation, then tuning exploration to serve that reward — differs from our setup: we ask whether degradation still depends on the exploration *setting itself*, holding the reward the agent is chasing constant.

RL scheduling on real spacecraft. NASA's Carruthers Geocorona Observatory mission (launched 24 September 2025; in orbit near the Sun–Earth L1 point, not LEO) uses a deep-RL (Maskable PPO) framework for long-horizon operations scheduling under power, thermal, and instrument constraints, including a cost term for "on-orbit degradation profiles" [CARRUTHERS]. In that work, degradation refers to instrument calibration responsivity drift (thermal variability and contaminant deposition affecting absolute photon response), not battery aging, and the paper does not study how the scheduler's exploration-exploitation setting affects that degradation term independently of science return. We cite it as evidence that deep RL is now trusted as the *default* operational scheduler for a real NASA mission, not as a LEO example and not as related measurement of the same quantity — the audience this paper's finding is relevant to already exists and is already flying RL schedulers, just not yet on a LEO, battery-cycling platform.

LEO battery-aging physics. [LEO-BATTERY-ARXIV] models battery aging for a 300-satellite, 550 km, 53°-inclination constellation (motivated by, though not identical to, SpaceX Starlink's scale) with a published log-linear cycle-life law, $\log_{10}(L) + \sigma d = \varepsilon$ (L = cycles to failure at constant depth d, σ a chemistry constant), and compares four scheduling heuristics (random, depth-of-discharge-first, min-power-deficit, min-net-energy-cost). It studies *which heuristic* minimizes aging under varying solar/thermal conditions; it does not study a learned (RL) policy or an exploration-exploitation parameter. We use its published parameters (altitude, battery capacity, SOC floor, and the aging law itself) as the physical basis for the second of our two simulation models (Section

4.2), specifically to test whether our finding — established on an unrelated toy model — survives under this independently-published aging law.

3. Problem Setup

3.1 Hypothesis

H: In an ε-greedy RL scheduler choosing among tasks with unknown value each orbit, the exploration coefficient ε drives total battery aging at fixed delivered mission value. That is, replaying one fixed task stream through the scheduler at several values of ε, and running each value until it reaches the same cumulative reward (not a fixed number of orbits), total battery aging rises monotonically in ε.

Falsified if: no monotone relationship between ε and total aging survives once cumulative reward is held constant across arms.

3.2 Why holding reward constant is the entire experiment

A naive comparison runs every ε for the same number of orbits and compares aging. This conflates two different things: more exploration means more energy spent on balance (lower average reward per orbit, since a random action is on average worse than the current best estimate), which by itself would use *more orbits* to deliver the same value even with zero aging-rate difference. An initial design in this line attempted to control for this with a partial correlation (aging vs. ε, controlling for achieved reward) and found ε and reward 97%-collinear (Spearman $\rho = -0.974$); a matched-reward subsample returned zero usable runs out of 96, because the reward ranges at different ε values did not overlap at all. A partial correlation computed under that degree of collinearity is not a control — there was no residual variance in reward left to condition on.

The design used here instead holds reward constant **by construction**: every (ε, seed) pair is run not for a fixed number of orbits, but until cumulative reward reaches a fixed target. The question becomes "how much battery damage did it cost to deliver this much mission value," which is answerable regardless of how collinear ε and reward are in the unconstrained case.

4. Method

Two independent simulation models are used, deliberately differing in orbit, battery scale, and the functional *form* of the aging law — not merely its constants — so that a result confirmed in both is not an artifact of one parameterization.

4.1 Scheduler (shared across both models)

An ε-greedy scheduler chooses among $N = 6$ tasks each orbit. With probability ε it picks a uniformly random task (explore); otherwise it picks the task with the highest current sample-mean value estimate (exploit). Task values are drawn once per model run from a fixed seed and replayed identically at every ε, so differences across ε are due to the scheduler, not to different tasks being presented. Twelve scheduler seeds are run per (model, ε) combination. At each orbit: the battery charges during the sunlit phase, a fixed housekeeping draw is subtracted, and — if a task is both chosen and energetically affordable without breaching the battery's floor — its energy cost is subtracted and its (noisy) observed value accumulates into the running reward and the task's running value estimate. Depth of discharge for the orbit is recorded; each simulation runs until cumulative reward reaches a fixed target (reward-matched design, Section 3.2), or a large orbit cap is hit.

4.2 Model 1: toy parameterization, power-law (Wöhler) aging

400 km circular orbit (92.4 min period, 15.58 cycles/day fixed by orbital mechanics — not a scheduler-controllable quantity), an arbitrary 100 Wh battery, 42 Wh recovered per sunlit phase, 14 Wh fixed housekeeping draw per orbit, task energy costs drawn uniformly in [4, 16] Wh. Per-cycle damage is modeled as DoD^1.87 (Wöhler/Palmgren-Miner, a standard Li-ion exponent), summed (Miner's rule) over all cycles to the reward target.

4.3 Model 2: constellation-scale parameterization, log-linear cycle-life aging

Parameters from [LEO-BATTERY-ARXIV]: 550 km altitude, 53° inclination (period and eclipse fraction derived here from the altitude via standard orbital mechanics — 95.50 min, 15.08 cycles/day, 37.2% eclipse at the $\beta = 0$ maximum-eclipse geometric bound — rather than copied from the source's rounded figures); 1200 Wh battery with a hard 20% minimum state-of-charge floor (a task that would breach the floor is skipped that orbit, not merely penalized); 75 W housekeeping draw (midpoint of the source's stated 50–100 W range). Per-cycle damage is modeled as $10^{\wedge}(\sigma d)$ with $\sigma = 0.8$, derived from the source's log-linear cycle-life relationship $\log_{10}(L) + \sigma d = \varepsilon$ by taking damage-per-cycle $\propto 1/L(d)$; the battery-specific constant ε is fixed to an arbitrary constant. This is exact, not approximate: total aging for an arm is $10^{(-\varepsilon)\cdot\Sigma_i 10}(\sigma d_i)$, so for any two arms (e.g. $\varepsilon=1$ vs. $\varepsilon=0$, same battery, same fixed ε-constant) the ratio of their totals is $10^{\wedge}(-\varepsilon)\Sigma_1 / 10^{\wedge}(-\varepsilon)\Sigma_0 = \Sigma_1/\Sigma_0$ — the unknown constant cancels identically, by algebra, regardless of its real value. No claim in this paper (all of which are ratios or rank statistics) depends on it. Charge-per-sunlit-phase and task energy costs are scaled from Model 1 by the ratio of battery capacities (12×), since the source gives wide uniform ranges for solar panel area and efficiency rather than a single representative satellite, and preserving Model 1's ratio-to-capacity is at least an explicit, checkable choice rather than an arbitrarily chosen point in those ranges.

4.4 What is, and is not, being validated against a real system

Neither model is a specific satellite. Model 2 uses a named operator's actual orbital geometry (Starlink Shell 1's altitude and inclination) and a battery scale and SOC floor from a paper modeling a comparably-sized constellation, under an aging law from an independent source — this is the strongest validation available without access to a real operator's battery chemistry, task manifest, or scheduler, none of which are public. The purpose of Model 2 is to test whether the qualitative finding from Model 1 survives a change in every physical input *and* a change in the functional form of the aging law, not to claim a measurement of any deployed system.

A property of both models worth stating plainly: the orbit-to-target sequence (Table 5.1 vs. Table 5.2, "orbits to target" column) is numerically identical between the two models at every ε. This is not a copy-paste artifact; it is a consequence of the battery floor rarely binding in either model — the charge recovered per sunlit phase is generous enough relative to task-energy draws that task success is governed almost entirely by the (unscaled) value estimates and the ε/seed-driven choice sequence, not by the physical energy scale. Task *values* are not rescaled between models (Section 4.3), so that choice sequence is driven by the same numbers in both.

This means Model 2 should be read precisely: it isolates the effect of the **aging law and battery scale** on the depth/count decomposition (Section 5.2–5.3), holding the scheduler's decision trajectory fixed. It is **not** an independent re-test of whether the orbit-count effect itself survives a different energy-constraint regime — that would require a third model where the floor binds often enough to change which tasks succeed. The claim that the two models are "structurally different" is accurate for the aging law and physical scale, and should not be read as implying the scheduler dynamics were independently re-randomized.

4.5 Initialisation of the greedy baseline, and its sensitivity

A greedy agent whose value estimates all start at zero has no basis for its first choice: `argmax` over tied zeros returns the first task, every run. If that task is not the best arm, $\varepsilon=0$ can stay stuck on it, because nothing else is sampled unless its own running estimate turns negative, which rarely happens for positive-valued rewards. A comparison against such a baseline measures the baseline's initialisation, not exploration.

All results therefore use a forced warm start: every task is tried once, in random order, before the ε-controlled policy takes over. It is applied identically at every ε, so it cannot favour the hypothesis. To measure how much it matters, each script also runs with `--no-warmstart`, which reproduces the naive initialisation. Over the same 30 task-value draws, $\varepsilon=0$ then aged the battery more than $\varepsilon=1$ in 14/30 draws for Model 1 (median ratio 1.04×, Wilcoxon $p = 0.34$) and 13/30 for Model 2 (1.04×, $p = 0.32$), and wore the actuator more in 19/30 draws for Model 3 (median 0.73×, $p = 0.81$). Without the warm start there is no detectable effect; with it, the effect is large and consistent. The result is therefore conditional on a fairly initialised greedy baseline, and a reader implementing the comparison should check their own baseline's initialisation first.

4.6 Model 3: generalizing off battery cycling to actuator wear

The paper's own Related Work (Section 2) raised, and left untested, whether the mechanism extends past battery cycling to mechanical actuator wear — the RL-robotics literature's concern, but never measured against delivered task value. Model 3 tests this directly, using real hardware parameters from "The Open Ant" [ROBOTICS-WEAR], fetched 3 Oct 2026: Dynamixel servo torques (0.28–0.82 Nm across the paper's Lite and full configurations) and documented joint angular ranges (20–70°). Per-action wear follows **Archard's law** — proportional to torque × angular displacement, a mechanical-work term with no "depth" concept at all — structurally distinct from both battery aging laws (Wöhler's power law and the log-linear cycle-life law). A separate thermal budget gates action affordability, motivated by the paper's own documented 72°C stall cutoff, with heat generated modeled as proportional to torque^2 (resistive heating scales with current^2, and current scales with torque at fixed voltage) — a different torque dependence from the wear law's linear one, so the gating resource and the measured outcome are not simply rescaled copies of each other. Same ε-greedy scheduler, same reward-matched design, same warm-start fix as Models 1–2.

5. Results

All results below use the warm start (Section 4.5) and report the 30-independent-draw robustness check as the primary evidence. A single-draw table is given for each model for readability, with the explicit caveat that a single draw is not, by itself, evidence of anything.

5.1 Model 1 (400 km, Wöhler)

Single-draw illustration: ε=0 reaches the reward target in 174.5 orbits (aging 11.19); ε=1 in 283.2 orbits (aging 20.54) — a 1.84× ratio on this one draw. **Across 30 independent task-value draws: 0/30 reversed.** Median ratio (total aging, ε=1 vs. ε=0) **1.47×**, Wilcoxon signed-rank (ratio > 1) $p = 9.3\times10^{-10}$.

5.2 Model 2 (550 km, log-linear cycle life)

Single-draw illustration, same task-value draw as Model 1 (orbit counts match exactly — the artifact Section 4.4 already discusses: the thermal/energy floor rarely binds in either model, so the choice *sequence* is driven by the same unscaled task values in both, though the aging *magnitude* differs because the aging law does): ε=0 reaches the target in 174.5 orbits (aging 269.7); ε=1 in 283.2 orbits (aging 453.7) — a 1.68× ratio on this one draw. **Across 30 independent task-value draws: 0/30 reversed.** Median ratio **1.58×**, range [1.15×, 2.02×], Wilcoxon $p = 9.3\times10^{-10}$.

5.3 Model 3 (Open Ant actuator, Archard wear)

Across 30 independent task-value draws: 2/30 reversed (ratio as low as 0.83×) — the first model, of three, where reversals remain with the warm start in place. Median ratio over all 30 draws **1.71×** (1.72× over the 28 where the effect holds); full range **[0.83×, 3.30×]**. Wilcoxon signed-rank across all 30 (ratio > 1): $p = 9.3\times10^{-9}$ — significant overall, but visibly less uniform than either battery model. A plausible reason: Archard's law is linear in torque × displacement with no saturating regime, so the balance between "more actions taken" and "more wear per action" is more sensitive to exactly which tasks get explored than the battery models' superlinear (Wöhler) or exponential (log-cycle-life) aging laws, both of which give the "more cycles" channel more room to dominate consistently.

5.4 What transfers and what does not

The *direction* transfers across all three models, but **not uniformly** — this is itself a finding, not a caveat to bury. Both battery models are fully robust (0/30 reversals each, $p = 9.3\times10^{-10}$); the actuator generalization is real but has a genuine, non-trivial exception rate (2/30, $p = 9.3\times10^{-9}$). The specific multiplier (1.47–1.71×) does not transfer and should not be quoted as a fixed number — it is a property of each physical model, not of the scheduler mechanism. The claim that survives all three models, stated at the level of precision the evidence supports: **exploration drives physical wear independently of delivered mission value, through both more**

total actions and more wear per action, with the action-count channel usually the larger contributor — not a specific ratio, and not "through depth/intensity alone," which all three models refute.

6. Discussion

Operational implication. Exploration rate in an onboard RL scheduler is ordinarily tuned purely for learning speed — how quickly the agent's value estimates converge. This result adds a second axis to that tuning decision: a higher exploration rate also costs battery life faster, at matched mission output, in a way that is measurable and — because it is a scheduler hyperparameter, not a hardware property — directly controllable without any hardware change. For a mission where battery life is the binding constraint on mission duration (true for many small LEO platforms), this may warrant considering exploration schedules that decay faster, or are capped lower, than a pure learning-speed objective alone would recommend — trading a slower-converging policy for a longer-lived battery. That recommendation is offered at the strength three simulations support: a design consideration to weigh, not a tuned default to adopt.

Why this is not already solved by existing degradation-aware RL. The EV-charging literature already incorporates degradation into the reward function, which in principle lets an RL agent learn to avoid degrading actions on its own. That approach requires the degradation model to be known and differentiable (or otherwise learnable) at design time, and folds the tradeoff into a single scalar reward the designer must still weight. The finding here is upstream of that design choice: it says the exploration *setting itself* — a hyperparameter present regardless of how the reward is shaped — has an aging cost, which matters even to a designer who has not yet built degradation into the reward at all, and gives a cheap first lever (ε) to pull before investing in a full degradation-aware reward redesign.

7. Limitations

This is a simulation study, run three times, not a flight-telemetry or real-hardware analysis. The task streams are synthetic (randomly drawn values and energy/wear costs), not a real satellite's or robot's mission manifest.

Baseline initialisation is a measured risk, not a hypothetical one (Section 4.5). Without the warm start, the same bandit difficulty used throughout reverses the headline relationship in 13–19 of 30 task-value draws. The warm start removes that specific artifact and is applied identically at every ε, but it does not establish that no other initialisation-sensitive artifact remains — only that this measured one does not.

Model 2's charge-per-orbit and task-energy parameters are scaled from Model 1 by a documented ratio rather than independently sourced, because the literature parameter this study drew on gives wide ranges rather than a single representative satellite. The battery-specific constant in Model 2's aging law is fixed arbitrarily because its real value is chemistry-specific and unpublished in the source; every comparison in this paper is a ratio or rank statistic that is invariant to that choice, but no absolute aging magnitude is claimed. Model 3's thermal-heating scale (torque^2) and wear coefficient are likewise unitless constants chosen once before running, not independently sourced, for the same reason. The literature check behind Section 2 is not a systematic review — four targeted search sessions and full-text checks of the closest candidates, not an exhaustive database search. A flight-data replay, or a physical robot replay for Model 3, would be the next tier of evidence beyond what this paper provides.

8. Conclusion

The exploration coefficient of an onboard RL scheduler is not only a learning-speed hyperparameter; it is also an independent driver of physical wear, measured at fixed delivered mission value and confirmed robust — fully for battery aging (two models, 0/30 reversals each), substantially for mechanical actuator wear (2/30 reversals) — across three simulation models differing in orbit, hardware scale, and the physical law governing damage. The effect holds only against a fairly initialised greedy baseline (Section 4.5), and is evidenced by 30 independent environments per model rather than one. The magnitude is model-dependent and not claimed to transfer to any specific deployed system; the qualitative design implication — exploration has a wear cost orthogonal to its learning-speed benefit, controllable via the same hyperparameter that controls learning speed, and real across

more than one kind of hardware — does transfer, and is offered here as a testable, falsifiable design consideration for the (already-flying) practice of RL-based spacecraft scheduling.

Declarations

Competing interests. The author declares no competing interests.

Use of AI assistance. The research question, the reward-matched experimental design, the decision to treat the baseline-initialisation issue as a reported sensitivity result, and all conclusions are the author's. An AI assistant (Claude, Anthropic) helped draft the text, write and run the simulation code, and check references, and the author reviewed the results.

Reproducing

Code, stored results and notes: https://github.com/AegisMindApp/exploration-cost-rl-wear (MIT licence; requires only numpy and scipy).

```
python run_assay.py                              # Model 1 (4.2, 5.1)
python run_assay_starlink_scale.py               # Model 2 (4.3, 5.2)
python run_assay_actuator_wear_generalization.py # Model 3 (4.6, 5.3)
# add --no-warmstart to any of the three for the Section 4.5 sensitivity run
```

All three are deterministic given their fixed seeds and run the 30-independent-draw robustness check as their primary output; the numbers in Section 5 were re-derived from these scripts. See `notes/RESULT.md`, `notes/STARLINK_SCALE_RESULT.md` and `notes/ACTUATOR_WEAR_RESULT.md` in that repository for the full record, including earlier single-draw results that are superseded.

References

- [ROBOTICS-WEAR] Lupu, E. S., Spieler, P., Javed, K., De Asis, K., & Martin, J. D. "The Open Ant: A Robot Platform for Reinforcement Learning Research." *Reinforcement Learning Journal* (2026). arXiv:2607.18488. Cited in Section 2 for its framing of physical RL platforms as needing frequent repair from real-world exploration; also the source of Model 3's real actuator torque (0.28–0.82 Nm) and joint angular range (20–70°) parameters (Section 4.6).
- [EV-CHARGING-RL] Haghighi, R., Hassan, A., Bui, V.-H., Hussain, A., & Su, W. "Deep Reinforcement Learning-Based Optimization of Second-Life Battery Utilization in Electric Vehicles Charging Stations." arXiv:2502.03412 (2025). Broader survey context: PMC10755288, "Progress and summary of reinforcement learning on energy management of MPS-EV."
- [CARRUTHERS] Zhang, A., Craig, J., & Waldrop, L. "Deep RL for Fast Long-Horizon Operations Scheduling on NASA's Carruthers Geocorona Observatory Mission." *Proceedings of the International Conference on Automated Planning and Scheduling*, 36(1), 538–546 (2026). arXiv:2606.22159. Launch date (24 Sep 2025) and orbit (Sun–Earth L1, not LEO) per NASA's mission status page (science.nasa.gov/blogs/carruthers-geocorona-observatory) and eoPortal, not stated in this paper itself.
- [LEO-BATTERY-ARXIV] Zeng, L., Zhu, J., Wang, Z., Shi, Y., & Letaief, K. B. "Unseen Cost of Space Computing: Quantifying LEO Battery Aging via Physics-Driven Modeling." arXiv:2603.04372 (2026).
- [HEALTH-AWARE-RL] Jha, M. S., Weber, P., Theilliol, D., Ponsart, J.-C., & Maquin, D. "A Reinforcement Learning Approach to Health Aware Control Strategy." *Mediterranean Conference on Control and Automation (MED)*, IEEE (2019). arXiv:2010.09269 (2020) — related (RUL-tracking reward design for mechanical wear), cited for contrast rather than overlap.
- [SMOOTH-EXPLORATION] Raffin, A., Kober, J., & Stulp, F. "Smooth Exploration for Robotic Reinforcement Learning." *Proceedings of the 5th Conference on Robot Learning*, PMLR 164, 1634–1644 (2022). arXiv:2005.05719 (2020). Cited in Section 2 as the closest exploration-damage mitigation method; abstract checked, full text not read.

Five of the six references were resolved and claim-provenance-checked directly against arXiv API metadata and abstracts (not search-engine paraphrase) in an earlier review of this draft; [SMOOTH-EXPLORATION] was added on 4 Oct 2026 and checked against its arXiv abstract only.

σ=0.8 (the [LEO-BATTERY-ARXIV] aging-law constant Model 2 depends on) was extracted via an AI-mediated tool read of the source paper, not a human line-by-line read of its equations.. This is the one number in the paper taken on that basis rather than independently re-derived; everything else in Section 5 was reproduced from committed code during this review.