Grokking on Real Astronomical Data

Does Delayed Generalization Survive Contact With Reality?

Ish Sitotombe

July 2026

Contents

Grokking on Real Astronomical Data: Does Delayed Generalization Survive Contact With Reality?

Draft — every number below is traceable to a log file in the repo.

Abstract

We test whether grokking — the delayed transition from memorization to generalization first documented on synthetic algorithmic tasks — also occurs when training data is real, sparse, and subject to measurement noise, using a tiny transformer trained on measured solar-system orbital data (Kepler's third law). A naive experimental design (8 planets, one central mass) produces a misleading negative result: the model appears to fail to generalize to moons of the outer planets, but this is a data identifiability artifact (central mass never varies in training) rather than a learning failure. We diagnose and fix two successive confounds — a distance-range gap and a mass-range gap between training and test data — by adding a small number of additional real, verified astronomical bodies (Earth's Moon, Mars's moons, Uranus's major moons, Pluto and Charon) to training. With both confounds resolved, the final model (18 real

training bodies spanning 5 central masses, 60,000 training steps) not only fits its training data but correctly generalizes Kepler's third law to 12 real, held-out bodies across three previously-unseen gravitational systems (Jupiter, Saturn, Neptune), with errors mostly under 10%. Critically, this training run (seed=0) exhibits genuine grokking: the model reaches a near-perfect training fit by step ~3,600, but held-out accuracy does not converge until roughly step ~27,600 — a ~24,000-step delay between memorization and generalization, the textbook signature of grokking. A mechanistic check confirms this is not a coincidence of outputs: the model's locally estimated mass and distance exponents on the 12 held-out bodies (-0.448 ± 0.026 and 1.417 ± 0.044) closely match Kepler's true values (-0.5 and 1.5) and are tightly clustered across all three unseen systems. A second seed replicates the successful generalization and the mechanistic exponent finding, but shows a noisier, longer, multi-episode approach to that outcome rather than seed=0's clean single delay. Scaling this check to 100 independent seeds (via a validated PyTorch port run on cloud infrastructure) shows the 2-seed result was itself optimistic: only 29 of 100 seeds (29%) reach successful cross-system generalization within the 60,000-step budget — the other 71% do not converge to a generalizing solution in that time. Among the seeds that do succeed, held-out mechanistic exponents remain reliably close to Kepler's true values (mass -0.43 ± 0.04, distance 1.37 ± 0.05 vs. true -0.5/1.5), so when generalization happens, it reflects real recovered physics rather than a fluke. We conclude that grokking-style delayed generalization does survive contact with real, noisy, non-synthetic data and, when successful, reflects genuine recovery of the underlying physical law — but success itself is seed-dependent and occurs in a minority of random initializations at this training budget, and the exact duration and shape of the pre-generalization delay should be treated as noisy rather than a precise, reproducible constant.

1. Introduction

Grokking — a model's abrupt transition from overfitting to generalizing long after training accuracy has saturated — was first documented by Power et al. (2022) on small synthetic algorithmic datasets such as modular arithmetic. Since then, the phenomenon has been studied extensively in similarly clean, synthetic settings.

An open question is whether grokking is an artifact of the unusually clean structure of these synthetic tasks, or whether it also emerges when the underlying data is real, sparse, and subject to measurement noise. This matters because most claims about grokking's theoretical implications (e.g. that models can be induced to discover "true" underlying laws) have only been demonstrated on data where a "true" law is exact by construction.

We use real orbital mechanics as a testbed because it has direct historical precedent: a small number of noisy, real measurements (planetary orbits) were historically sufficient for scientists to infer a general physical law (Newton's law of gravitation) and correctly extrapolate it to a new regime entirely (the moons of Jupiter, discovered by Galileo and later shown to obey the same law). That historical generalization required evidence spanning more than one gravitational system — planetary orbits around the Sun were not, by themselves, enough to identify how the law depended on the mass of the central body, since the Sun's mass never varied across those observations. We test whether a tiny transformer, trained on a literal handful of real measurements,

can recover the same relationship (Kepler's third law, a special case of Newton's law) and correctly generalize it to previously unseen gravitational systems — and we find that the same identifiability requirement applies to the model as applied historically to human scientists: a single-system training set is not enough, regardless of how well the model fits it.

Contributions: 1. We test grokking against real, noisy scientific measurement data rather than synthetic ground truth with a law that is exact by construction — to our knowledge an underexplored setting for the grokking literature. 2. We identify and diagnose two successive, real confounds that make a naive version of this experiment produce a misleading negative result — a training/test distance-range gap, and (after fixing that) a training/test central-mass-range gap — and show that fixing both with genuine additional real-world data (not synthetic augmentation) changes the outcome from apparent failure to a clear generalization success. 3. We show the successful run exhibits textbook delayed generalization (grokking): training fit converges roughly 24,000 steps before held-out accuracy does, directly answering this paper's title question. 4. We test the robustness of this finding across 100 independent random seeds and show a materially more nuanced picture than a single- or two-seed result would suggest: successful cross-system generalization occurs in only ~29% of seeds within this step budget, though when it does succeed, the recovered mechanistic exponents are reliably close to the true values.

2. Related Work

- Power et al. (2022) — original grokking paper, modular arithmetic.
- Nanda et al. — mechanistic account of the Fourier/circular algorithm learned for modular addition.
- Lee et al. (2024), "Grokfast: Accelerated Grokking by Amplifying Slow Gradients" (arXiv:2405.20233) — EMA gradient filter intended to speed up the transition to generalization.
- Thilak et al. (2022), "The Slingshot Mechanism: An Empirical Study of Adaptive Optimizers and the Grokking Phenomenon" (arXiv:2206.04817) — documents cyclic stable/unstable training regimes under adaptive optimizers + weight decay, causing periodic post-grok accuracy dips. Directly relevant: our Phase 0 baseline run shows this exact pattern (see 4.1).
- None of the above use real measured scientific data with the identifiability structure real data has (variables that are accidentally constant or non-overlapping across a training/test split) — this is the gap this paper addresses.

3. Method

3.1 Data

All orbital data is real, measured astronomical data (NASA/NSSDC/JPL sources, or standard well-documented IAU constants where no single NSSDC-hosted table covered a body) — no synthetic or fabricated values appear anywhere in the dataset. Full source citations and precise values are in `orbital_data.py`.

- **Phase 1 (training), final design:** 18 real bodies spanning 5 distinct cen-

tral masses — the 8 planets plus Pluto (central mass = Sun), Earth's Moon (central mass = Earth), Phobos and Deimos (central mass = Mars), Miranda/Ariel/Umbriel/Titania/Oberon (central mass = Uranus), and Charon (central mass = Pluto). Features: central mass (kg), orbital semi-major axis (km). Target: orbital period (days).

- **Phase 2 (held out, cross-system, never trained on), final design:** 12 real bodies across 3 gravitational systems never represented among the training central masses — the 4 Galilean moons of Jupiter, 7 major moons of Saturn (Titan, Enceladus, Iapetus, Rhea, Mimas, Tethys, Dione), and Triton (Neptune).
- **Preprocessing:** central mass and distance inputs, and the period target, are all log10-scaled before being fed to the model. This is necessary given the dynamic range involved (periods span 0.32 to ~90,562 days across the full dataset; central masses span ~13 orders of magnitude) and also matches the underlying physics, since Kepler's third law is linear in log-log space.
- **Train/test split:** no held-out split within Phase 1 — all 18 bodies are used for training. This was a deliberate choice (see changelog, 2026-07-04): with so few real bodies available, any in-domain holdout would be too small to support a meaningful conclusion on its own. The real generalization claim in this paper rests entirely on Phase 2 (the cross-system, cross-mass-regime held-out set), not on an in-domain train/test split.
- **Note on how this dataset reached its final form:** it was not designed correctly on the first attempt. The first version (8 planets, 1 central mass) and second version (11 bodies, 3 central masses) both produced misleading negative-looking results due to real, diagnosed confounds — see section 4.3 for the full account. The final 18-body, 5-mass design described above is what actually enabled successful generalization.

3.2 Model

Tiny decoder-only transformer: 2 layers, 4 attention heads, d_model=128, d_ff=512. Identical architecture and hyperparameters used for both the modular-arithmetic sanity check (Phase 0) and the real orbital-data experiments (Phase 1/2), to keep the comparison clean. For Phase 1/2, the token-embedding and softmax-classification layers used for modular arithmetic are replaced with a linear scalar-to-d_model projection (input: 2 scalar features per example, log-mass and log-distance) and a linear regression head (output: a single scalar, log-period) respectively — see `orbital_train.py` for the exact adaptation.

3.3 Training

Phase 0 (modular arithmetic): AdamW, weight decay=1.0 (default; weight decay is the primary lever known to induce grokking). LR=1e-3, batch size=512, train_frac=0.5 (4704 train / 4705 test examples for mod-97 addition). A weight_decay=0.1 run was also tested (see 4.1) and did not grok within budget, confirming weight decay strength matters as expected.

Phase 1/2 (real orbital data): AdamW, weight_decay=0.1, LR=1e-3, full-batch gradient descent every step (batch size = dataset size, since the entire training set is only

18 real examples). The final successful run used 60,000 steps; an earlier attempt at 20,000 steps (the same step budget that had sufficed for the smaller 8- and 11-body dataset versions) was insufficient for training loss itself to converge on the harder, more heterogeneous 18-body dataset (see 4.3) — this is a purely practical finding about how much optimization budget a richer real dataset needs, not a new confound in the data design.

Hardware for Phase 0-4.4 and the seed=0/seed=1 checks (4.5): Apple M1 MacBook Pro, MLX (Metal backend).

Seed-distribution study (4.5, n=20 and n=100): running many independent full training runs is embarrassingly parallel across seeds, but MLX is Apple-Silicon-only and the M1 is single-machine, so this study used a PyTorch port of the training code (`orbital_train_torch.py`), validated against the original MLX seed=0 result before being trusted at scale (final train/held-out MSE and held-out exponents matched closely — see changelog, 2026-07-05). Batches were run via `orbital_batch.py` (multiprocessing across seeds, one process per CPU core) on an AWS c7i.8xlarge instance (32 vCPUs, eu-west-1), each seed still trained full-batch for the same 60,000 steps as the MLX runs.

4. Experiments

4.1 Sanity check: modular arithmetic

Confirmed: the harness reproduces known grokking behavior.

- **Baseline (weight_decay=1.0, seed=0):** train accuracy reached ~100% by step ~500 (memorization). Test accuracy stayed near-random until it crossed 99% at **step 1300** — the expected flat-then-cliff shape. Log: `baseline_log.csv`, curve: `baseline_curve.png`.
  - After grokking, test accuracy did not settle into a flat plateau — it exhibited repeated dips back toward random (e.g. steps 2100, 2700, 3300, 3800, 4500, 4800, 9000, 15000, ...) before re-recovering, for the remainder of the 30,000-step run. This matches the "Slingshot Mechanism" (Thilak et al. 2022): cyclic stable/unstable regimes under AdamW + strong weight decay. We treat this as an expected, literature- consistent artifact of wd=1.0, not a bug in the harness.
- **weight_decay=0.1, seed=0, 10,000 steps:** did NOT grok — test accuracy only reached ~0.17-0.28 by step 10,000, confirming that weight decay strength is load-bearing as expected from the literature. Log: `baseline_wd01_log.csv`.
- **Grokfast (alpha=0.98, lamb=2.0, wd=1.0, seed=0, 5,000 steps):** test accuracy approached but did not cross the 0.99 threshold — reached 0.981 at step 3700, then hit a slingshot-style collapse at step 3800 and had not recovered by step 5000 (ended at 0.282). Notably, Grokfast did *not* outperform the plain baseline here (baseline crossed 99% at step 1300; Grokfast had only reached 0.981 by step 3700) — plausibly because wd=1.0 already groks fast enough that there's little headroom for the EMA gradient filter to accelerate, and/or the filter interacts with the slingshot dynamic to produce a larger collapse. Log: `grokfast_log.csv`.

- **Grokfast, matched comparison (alpha=0.98, lamb=2.0, wd=1.0, seed=0, 30,000 steps — same seed and step budget as the baseline):** grokked at **step 1900**, vs. the baseline's **step 1300**. Grokfast was *slower* to grok here, contrary to its intended effect. Both runs show the same slingshot-style oscillation for the remainder of training (e.g. Grokfast run: steps 4100, 11500, 14200, 14900, 15900, 22200 all show near-total collapses back to random before recovering). Logs: `baseline_log.csv` vs. `grokfast_30k_log.csv`.
- **Ablation to test the "no headroom" hypothesis (wd=0.3, seed=0, 15,000 steps, baseline vs. Grokfast):** baseline grokked at **step 2600**; Grokfast grokked at **step 5100** — 96% slower, a larger relative gap than at wd=1.0 (46% slower). This rules out the headroom hypothesis: if Grokfast were only failing to help because wd=1.0 already groks fast, giving it more room (via lower weight decay, where baseline itself is slower) should have let it close the gap or overtake baseline. Instead the gap widened. Logs: `baseline_wd03_log.csv` vs. `grokfast_wd03_log.csv`.
  - **Revised conclusion:** across two weight-decay settings, Grokfast (alpha=0.98, lamb=2.0) consistently and reproducibly *delayed* grokking rather than accelerating it, in this architecture/optimizer/ task combination. This is reported as a genuine negative result — no alpha/lamb sweep was run to search for a configuration that would reverse it, and we do not claim Grokfast is ineffective in general, only in this specific tiny-transformer/AdamW/wd-regularized setup. A plausible mechanism (untested): the EMA filter's fixed alpha may be poorly matched to the timescale of this task's memorization phase, causing it to amplify noise from the memorization gradient rather than the later generalizing signal — but this would need a proper alpha/lamb sweep to confirm and is flagged as future work, not fact.
- **Bonus ablation at the authors' recommended low weight decay (wd=0.05, seed=0, 20,000 steps):** baseline grokked at **step 19,400** (test acc crossed 0.99). Grokfast **never grokked** within the full 20,000-step budget — test accuracy stayed stuck near ~1% (0.010) for the entire run, while train accuracy hit 100% almost immediately. This is the strongest version of the pattern: at the exact hyperparameter regime the Grokfast authors recommend for arithmetic tasks, in our setup it didn't just delay grokking, it prevented it entirely within a budget where the baseline succeeded. Logs: `baseline_wd005_log.csv` vs. `grokfast_wd005_log.csv`.
  - **Final Phase 0 conclusion:** across three weight-decay settings (1.0, 0.3, 0.05), Grokfast (alpha=0.98, lamb=2.0) consistently underperformed the plain baseline in this architecture/optimizer/task combination — slower at high weight decay, and a complete failure to grok at the recommended low weight decay. Implementation verified against the official Grokfast reference code (github.com/ironjr/ grokfast) line-by-line; matches exactly. This is reported as a real, reproducible negative result specific to this configuration, not a general claim that Grokfast doesn't work — the authors themselves note both hyperparameter sensitivity and an unresolved interaction with weight decay in their own paper.

4.2 Real data: planets around the Sun

Trained on all 8 planets (central mass = Sun, constant across all rows; see 3.1 design note), log10-space inputs/targets, wd=0.1, 20,000 steps, seed=0. Log: `orbital_baseline_log.csv`, curve: `orbital_baseline_curve.png`.

Train MSE (log10 period) dropped from 1.42 at step 1 to ~0.0001-0.0005 by step 20,000 — an essentially perfect fit. Predicted vs. true periods:

Planet	True period (d)	Predicted period (d)	Error
Mercury	87.97	93.73	+6.5%
Venus	224.70	230.91	+2.8%
Earth	365.26	379.79	+4.0%
Mars	686.98	732.81	+6.7%
Jupiter	4332.67	4555.66	+5.1%
Saturn	10759.35	11104.51	+3.2%
Uranus	30685.54	33612.86	+9.5%
Neptune	60190.53	60062.12	-0.2%

This is expected and unsurprising: with the Sun's mass held constant across all 8 rows, the task reduces to fitting period against distance alone — a low-noise, near-exact power law (Kepler's third law) from 8 real points. This confirms the harness can fit a real physical relationship from a handful of noisy measurements, but does not yet test the interesting question (see 4.3).

4.3 Cross-system generalization: moons of the outer planets

Evaluated (never trained on) against 9 moons of Jupiter, Saturn, and Neptune. Held-out MSE (log10 period) did not converge — it fluctuated between roughly 0.2 and 3+ throughout training and never approached the near-zero values seen on the training planets (see `orbital_baseline_curve.png`). Per-body predictions:

Moon (host)	True period (d)	Predicted period (d)	Ratio (true/pred)
Io (Jupiter)	1.77	1.38	1.3x
Europa (Jupiter)	3.55	1.82	1.9x
Ganymede (Jupiter)	7.15	2.44	2.9x
Callisto (Jupiter)	16.69	3.55	4.7x
Titan (Saturn)	15.95	2.66	6.0x
Enceladus (Saturn)	1.37	1.00	1.4x
Iapetus (Saturn)	79.33	5.61	14.1x
Rhea (Saturn)	4.52	1.58	2.9x
Triton (Neptune)	5.88	1.25	4.7x

The model **systematically and substantially underestimates** every moon's period, with the underestimate growing worse for moons further from the Sun-planet training

distribution (e.g. Iapetus, the most distant/longest- period moon tested, is off by 14x; Enceladus, closest to Sun-planet scale proportionally, is off by only 1.4x).

Why this happens — and why it is not primarily a grokking/architecture failure: in Phase 1, the central mass feature (the Sun's mass) is *identical* for all 8 training rows. There is therefore no gradient signal during training that could teach the model the correct coefficient for how period depends on central mass — that term is unidentifiable from single-mass data; its effective weight is absorbed into a bias term and can take on an arbitrary value without affecting Phase 1 training loss at all. When Phase 2 introduces genuinely varying central mass (Jupiter/Saturn/ Neptune, each far less massive than the Sun), the model was never in a position to have learned the correct mass-scaling relationship, regardless of training procedure, architecture, or how well it grokked the distance term. This is a data/identifiability limitation of the single-system training design, not evidence that grokking-style training fails to generalize scaling laws in general.

This also nuances the Newton/Kepler historical framing from the introduction: Newton did not derive gravity's mass-dependence purely from planetary orbit data with the Sun's mass held fixed either — he unified it with terrestrial gravity (falling bodies) and multiple independent lines of evidence, not statistical inference from single-central-mass orbital data alone. The current experimental design as run cannot recreate that step, and this paper reports that limitation explicitly rather than reframing the failure as a more dramatic generalization result than it is.

Redesign (2026-07-05), following the above finding: to distinguish "didn't learn the mass dependence" from "can't extrapolate past the trained distance range" (a second, independent confound — every Phase-2 moon distance falls below the *entire* planetary training distance range, by more than an order of magnitude with zero overlap: planets span log10(distance_km) ≈ 7.76-9.65, moons span ≈ 5.38-6.55), Earth's Moon and Mars' two moons (Phobos, Deimos) were added to **Phase 1 training**. These are real bodies with central masses other than the Sun (Earth, Mars), and Earth's Moon's distance (384,400 km, log10 ≈ 5.585) falls inside the Phase-2 test range. Phase 1 training set is now 11 real bodies across 3 central masses instead of 8 bodies across 1. Phase 2 (Jupiter/Saturn/ Neptune's large moons) is unchanged — still fully held out. See `orbital_data.py` for the updated dataset and design note.

Re-run result: train MSE converged to near-zero (all 11 train bodies fit within ~1-3% of true period, see `orbital_v2_log.csv`), but held-out MSE still did not converge — it oscillated around ~0.6 for the entire 20,000-step run, and every held-out prediction collapsed toward ~27-29 days regardless of the moon's true period (which ranges 1.4-79 days). This is a *different* failure mode than the first run (which systematically underestimated), and pointed to a second, unaddressed confound.

Root cause identified: the SAME kind of gap recurred, but in mass-space rather than distance-space. Training central masses (log10 kg) were Mars 23.81, Earth 24.78, Sun 30.30 — a 5.5-order-of-magnitude gap with nothing trained in between. Every held-out central mass (Neptune 26.01, Saturn 26.75, Jupiter 27.28) falls squarely inside that untrained gap — the model had never seen a central body anywhere near that mass regime, so it collapsed toward its nearest training anchor (Earth's Moon, 27.32 days) rather than extrapolating the law.

Second redesign (2026-07-05): real bodies whose central masses fill the gap were added to Phase 1 training: Uranus's 5 major moons (Miranda, Ariel, Umbriel, Titania, Oberon — central mass = Uranus, log10 = 25.94, landing right in the gap) and Charon (central mass = Pluto, log10 = 22.11, extending the low end below Mars). Pluto itself was added as a 9th Sun-orbiting "planet" for completeness, since it now supplies Charon's central body. Phase 1 training is now 19 real bodies across 5 central masses (Sun, Earth, Mars, Uranus, Pluto). Saturn's remaining major moons (Mimas, Tethys, Dione) were added to **Phase 2** rather than training, since Saturn is a held-out test system — this makes that part of the generalization test bigger and more robust (7 Saturn moons instead of 4) without leaking test-system data into training. Phase 2 is now 12 bodies across Jupiter, Saturn, and Neptune. See `orbital_data.py` DESIGN NOTE 2 for full detail.

Re-run result, 20k steps (`orbital_v3_log.csv`): held-out error improved substantially over the previous run (no longer collapsed to a single constant value — predictions now vary meaningfully with each moon's actual mass/distance), but a new confound appeared: training itself had not converged. Train MSE only reached ~0.01 by step 20,000 (vs. ~0.0001-0.0005 in earlier, smaller-dataset runs) and was still visibly oscillating rather than settling — expected, since 18 points spanning 5 central masses and ~8 orders of magnitude in mass is a substantially richer function to fit than the earlier near-1D curve, and the same wd=0.1/20k-step budget that sufficed before hadn't given this harder problem enough room to converge.

Re-run, extended to 60k steps (`orbital_v3_long_log.csv`), same seed/wd/lr: train MSE converged to ~0.0003-0.0004, held-out MSE converged to a closely matched ~0.0003-0.0008 — i.e. the model no longer performs meaningfully worse on unseen systems than on the ones it trained on. Per-body held-out predictions, all within ~12% of true period:

Moon (host)	True period (d)	Predicted period (d)	Error
Io (Jupiter)	1.7691	1.9625	+10.9%
Europa (Jupiter)	3.5512	3.7648	+6.0%
Ganymede (Jupiter)	7.1545	7.3665	+3.0%
Callisto (Jupiter)	16.6890	16.6661	-0.1%
Titan (Saturn)	15.9454	15.6932	-1.6%
Enceladus (Saturn)	1.3702	1.4787	+7.9%
Iapetus (Saturn)	79.3302	71.7181	-9.6%
Rhea (Saturn)	4.5175	4.5792	+1.4%
Triton (Neptune)	5.8768	5.8521	-0.4%
Mimas (Saturn)	0.9424	1.0559	+12.0%
Tethys (Saturn)	1.8878	1.9889	+5.4%
Dione (Saturn)	2.7369	2.8258	+3.2%

This is a genuine positive cross-system generalization result. A tiny transformer, trained on 18 real, noisy solar-system measurements spanning 5 central masses (Sun, Earth, Mars, Uranus, Pluto), correctly extrapolated Kepler's third law to 12 real bodies across 3 gravitational systems (Jupiter, Saturn, Neptune) it never saw

during training — with error magnitudes comparable to the training fit itself, not the systematic, scale-dependent collapse seen in the two earlier (confounded) attempts. Both confounds diagnosed along the way — zero distance-range overlap, then zero mass-range overlap — were real and had to be fixed with actual additional real-world data before this result was trustworthy; neither fix involved synthetic or fabricated data points.

This run also shows genuine, textbook grokking — not just an eventual correct answer, but the specific delayed-generalization shape the term refers to. From `orbital_v3_long_log.csv`: train MSE first reaches a near-perfect fit (0.00016) at **step 3,600**, while held-out MSE at that same step is still 0.45 — no better than random guessing in this log-scaled loss. Held-out MSE does not drop into the same near-zero regime (below ~0.001, matching the training fit's own scale) until roughly **step 27,600** (0.00047), with the transition beginning around step 21,000-21,200 where held-out MSE first crosses below 0.03. That is a **delay of roughly 18,000-24,000 steps** between the model solving its training data and solving held-out data it had never seen — the exact "flat-then-cliff" shape documented in the original grokking literature (Power et al. 2022) and reproduced on synthetic modular arithmetic in this paper's own Phase 0 (section 4.1). This is the direct, empirical answer to this paper's title question: delayed generalization does survive contact with real, noisy, non-synthetic measurement data, at least in this setting, once the training data is designed so the target relationship is actually identifiable from it.

4.4 Mechanistic analysis

Ran via `orbital_train.py --extract_exponents` (same 60,000-step training run as 4.3, seed=0). Method: at each of the 18 training points and 12 held-out points, the log-mass and log-distance inputs were each nudged by a small amount (central finite difference, h=1e-3) and the resulting change in the model's log-period output was measured. If the model implements Kepler's third law (period ∝ mass^-0.5 × distance^1.5), these local slopes should cluster tightly around -0.5 (mass) and 1.5 (distance) everywhere — training and held-out alike. Full per-body values in `orbital_v3_exponents_exponents.csv`.

Split	mass exponent (true: -0.5)	distance exponent (true: 1.5)
Train (18 bodies)	-0.4006 ± 0.1111 [-0.5032, -0.1302]	1.4345 ± 0.1401 [1.0605, 1.6632]
Held-out (12 bodies)	**-0.4481 ± 0.0259 [-0.4988, -0.4055]**	**1.4166 ± 0.0439 [1.3307, 1.4865]**
All 30 bodies	-0.4196 ± 0.0906	1.4273 ± 0.1124

The held-out numbers are the ones that matter for this paper's claim, and they are both close to the true values and tightly clustered (std ≈ 0.03-0.04, vs. Kepler's true exponents 0.5 and 1.0 away from zero respectively) across all 12 bodies spanning 3 independent gravitational systems — this is not a coincidence of fitting a handful of points, since these bodies were never used to fit anything. This

confirms the earlier positive result (4.3) mechanistically: the model isn't just numerically close to correct on these 12 outputs, it computes outputs consistent with actually implementing something close to the true two-body law locally around each of them.

Training-set exponents are noticeably noisier, particularly for Neptune (-0.1693) and Pluto (-0.1302), whose local mass-exponent estimates are far from -0.5. This has a plausible, non-alarming explanation rather than undermining the finding: 9 of the 18 training points (the Sun-orbiting planets and Pluto) all share the exact same central mass (the Sun), so locally around any single one of them, the training data offers very little actual variation in the mass direction to pin down that partial derivative precisely — the model can fit the training loss just fine while the mass-direction slope specifically near those points is under-constrained. The training bodies with real local mass variation nearby (Uranus's moons, Earth's Moon, Deimos) show mass exponents much closer to -0.5 (e.g. Titania -0.5032, Oberon -0.5016, Moon -0.4964, Deimos -0.4968). The held-out moons' central masses (Jupiter, Saturn, Neptune, log10 ≈ 26.0-27.3) sit closer to this well-constrained region of the training data (Uranus and Charon, log10 ≈ 22.1-25.9) than to the far tail occupied by the Sun alone — plausibly why their local slope estimates come out tighter and closer to the true value than the Sun-orbiting training planets' do.

4.5 Seed robustness check

Section 7 flagged single-seed results as a limitation. Re-ran the identical final design (18 training bodies, 12 held-out bodies, 60,000 steps, wd=0.1, lr=1e-3) with **seed=1** instead of seed=0 (`orbital_v3_seed1_log.csv`, `orbital_v3_seed1_exponents.csv`).

What replicates: cross-system generalization still succeeds. Held-out errors are mostly under 5% (e.g. Europa +0.6%, Triton +0.1%, Rhea -0.9%), with the same body — Iapetus — as the worst outlier in both seeds (-9.6% at seed=0, -29.5% at seed=1), plausibly because it is the most extreme distance-outlier among Saturn's moons (semi-major axis 3.56M km, vs. 185K-1.22M km for the others), making it the hardest single extrapolation case regardless of seed. The mechanistic exponents also replicate the qualitative pattern: held-out mass exponent -0.4698 ± 0.0138 and distance exponent 1.3982 ± 0.0691 (true: -0.5, 1.5) — again tightly clustered and close to true values, again noticeably tighter than the training-set exponents (-0.3457 ± 0.1568 mass, 1.4474 ± 0.1120 distance), matching seed=0's pattern that held-out estimates are more reliable than train-set ones near the Sun-mass cluster.

What does not replicate cleanly: the specific, clean "single-cliff" shape of the grokking delay. This run is substantially noisier throughout. Train MSE flickers to near-zero values as early as step ~6,600-7,400, but does not settle predominantly low until past step ~37,600 — over 30,000 steps later, itself an oscillatory, multi-episode process rather than one clean drop. Held-out MSE follows a similarly drawn-out path, and both metrics suffer a substantial, temporary relapse together around steps 45,000-46,600 (train MSE back up to 0.03-0.15, held-out MSE up to 0.09-0.21) before recovering by the end of training. The precise "~24,000-step delay" figure reported for seed=0 (4.3) is therefore **not a fixed property of this setup** — it was, at least in part, an artifact of which particular noisy trajectory seed=0 happened to take. What does hold up across both seeds is the qualitative claim that (a) a substantial

delay of many thousands of steps exists between early signs of a good training fit and reliable held-out performance, and (b) the final, settled outcome (once training is run long enough) is successful generalization with mechanistically-confirmed exponents. The exact shape and duration of that delay is seed-sensitive and should not be over-interpreted from a single run.

Scaling to many seeds (n=20, then n=100). The seed=1 comparison above showed a *qualitative* difference in delay shape but did not question whether successful generalization itself was seed-robust — with 2/2 seeds succeeding, that appeared to be a given. It wasn't. A batch of 20 more seeds (`orbital_batch.py`, PyTorch port, seeds 0-19) found only **8/20 (40%)** reached successful generalization (final held-out MSE < 0.01) within the 60,000-step budget. A follow-up batch of 100 seeds (`orbital_batch_100.csv`, seeds 0-99) narrowed this further to **29/100 (29%)** — the success rate was still moving between n=20 and n=100, so 29% should be read as a more precise but not necessarily fully converged estimate, not an exact figure.

Among the 29 seeds that succeeded: held-out mass exponent -0.4320 ± 0.0426 (true: -0.5); held-out distance exponent 1.3692 ± 0.0486 (true: 1.5) — consistent with both the seed=0 and seed=1 spot checks above, so the mechanistic finding (successful runs recover something close to the true physics) holds up well at scale. Grokking delay among the successes: mean 2772, stdev 4040, min -200, max 13200, using a mechanical threshold-crossing heuristic (train MSE first below 0.01, then held-out MSE first below 0.05) that is *not* directly comparable to the tighter, manually-inspected thresholds used to report seed=0's "~24,000-step" figure in 4.3 — the min of -200 is a heuristic artifact (held-out MSE crossed its threshold at the same or an earlier checkpoint than train MSE did in that one run), not a real "generalized before memorizing" event.

Revised headline finding. Across 100 independent seeds, under this exact architecture/hyperparameter/step-budget configuration, successful cross-system generalization occurs in a minority of runs (~29%), not reliably. When it does occur, the model's implicitly learned exponents are consistently close to Kepler's true values. This materially changes this paper's claim from "grokking-style generalization reliably survives contact with real data" (which the 2-seed check alone would have supported) to "grokking-style generalization is *possible* on real data and, when it succeeds, recovers correct physics — but success itself is seed-dependent and occurs in well under half of random initializations at this training budget." Whether a longer step budget would raise the success rate — i.e. whether the 71% of "failing" seeds are stuck rather than just slower — was not tested and is flagged as the natural next question, not something this paper claims to answer.

5. Results

Summary of all key numbers, each traceable to a log file listed at the end of this document.

Experiment	Metric	Value
Modular arithmetic sanity check (wd=1.0)	Step of grokking	1300
Modular arithmetic (wd=0.1, 10k steps)	Grokked?	No (test acc ~0.17-0.28)
Modular arithmetic (Grokfast, wd=1.0, 5k steps)	Step reached 0.981 test acc / grokked?	Step 3700 / not confirmed grokked (collapsed before recovering)
Modular arithmetic (Grokfast, wd=1.0, 30k steps, matched to baseline, seed=0)	Step of grokking	**1900 (46% slower than baseline's 1300)**
Modular arithmetic (baseline, wd=0.3, 15k steps, seed=0)	Step of grokking	2600
Modular arithmetic (Grokfast, wd=0.3, 15k steps, seed=0)	Step of grokking	**5100 (96% slower than wd=0.3 baseline's 2600)**
Modular arithmetic (baseline, wd=0.05, 20k steps, seed=0)	Step of grokking	19,400
Modular arithmetic (Grokfast, wd=0.05, 20k steps, seed=0)	Step of grokking	**Never grokked (stuck ~1% test acc for full 20k steps)**
Real planets (Phase 1, train fit, 8-body/1-mass design)	Final train MSE (log10 period)	~0.0001-0.0005 (near-perfect fit, all planets within ~10%)
Cross-system (moons, Phase 2, held out, 8-body/1-mass design)	MSE (log10 period) / per-body error	Did not converge; systematic underestimate, 1.3x-14.1x depending on distance from training scale
Cross-system (moons, Phase 2, held out, 11-body/3-mass design)	MSE (log10 period) / per-body error	Did not converge; predictions collapsed to ~27-29 days regardless of true period (1.4-79 days) — new confound (mass-range gap) identified
Real bodies, final design (Phase 1, train fit, 18-body/5-mass design, 60k steps)	Final train MSE (log10 period)	~0.0003-0.0004 (near-perfect fit)

Experiment	Metric	Value
Cross-system (moons, Phase 2, held out, final 12-body/3-system design, 60k steps)	MSE (log10 period) / per-body error	**Converged to ~0.0003-0.0008 — all 12 held-out moons predicted within ~12% of true period (most within 5%)**
Real orbital data (final design, 60k steps)	Delay between train convergence and held-out convergence	**~18,000-24,000 steps (train near-perfect by step 3,600; held-out doesn't converge until ~step 27,600) — textbook grokking, on real data**
Mechanistic (held-out bodies, 12 total, 3 systems, seed=0)	Locally estimated mass exponent (true: -0.5)	**-0.4481 ± 0.0259 [-0.4988, -0.4055]**
Mechanistic (held-out bodies, 12 total, 3 systems, seed=0)	Locally estimated distance exponent (true: 1.5)	**1.4166 ± 0.0439 [1.3307, 1.4865]**
Seed robustness (seed=1, same design, 60k steps)	Held-out exponents	Mass -0.4698 ± 0.0138, distance 1.3982 ± 0.0691 (true: -0.5, 1.5) — replicates
Seed robustness (seed=1)	Grokking delay shape	Does NOT cleanly replicate — noisier, multi-episode (train unsettled until ~step 37,600; large train+held-out relapse at steps 45,000-46,600) rather than seed=0's single clean cliff
Seed robustness (n=100, PyTorch port, 60k steps each)	Success rate (final held-out MSE < 0.01)	**29/100 (29%)** — successful generalization occurs in a minority of random seeds at this step budget
Seed robustness (n=100, among 29 successes)	Held-out exponents	Mass -0.4320 ± 0.0426, distance 1.3692 ± 0.0486 (true: -0.5, 1.5) — consistent with seed=0/seed=1 spot checks

Experiment	Metric	Value
Seed robustness (n=100, among 29 successes)	Grokking delay (mechanical threshold heuristic, not directly comparable to seed=0's manual 4.3 figure)	mean 2772, stdev 4040, min -200 (heuristic artifact), max 13200

6. Discussion

The path to this paper's headline result ran through two distinct, diagnosable confounds, each real and each fixed with additional real data rather than by reframing the failure:

1. **Distance-range confound (first design, 8 bodies, 1 central mass).** Training only on the 8 planets meant central mass (the Sun's) never varied, so the mass term of Kepler's third law was statistically unidentifiable — its effective coefficient could take any value without affecting training loss. Compounding this, every Phase-2 moon's distance fell below the entire training distance range, with zero overlap. The model's systematic, distance-scaled underestimation of moon periods was consistent with both confounds, not evidence against grokking-style training per se.
2. **Mass-range confound (second design, 11 bodies, 3 central masses).** Adding Earth's Moon, Phobos, and Deimos fixed the distance-overlap problem and gave the model varying central mass for the first time — but the specific masses used (Earth, Mars, Sun) left a 5.5-order-of- magnitude gap that happened to contain every Phase-2 test mass (Jupiter, Saturn, Neptune). Held-out predictions collapsed toward a near-constant value, consistent with extrapolating from the nearest training anchor rather than the general law.

Final design (18 bodies, 5 central masses: Sun, Earth, Mars, Uranus, Pluto), once training was run long enough to actually converge (60,000 steps): the model correctly predicted the orbital period of all 12 held-out bodies across three previously-unseen gravitational systems (Jupiter, Saturn, Neptune), with error magnitudes (mostly under 10%, worst case 12%) comparable to its own training fit. This is a genuine positive result: a tiny transformer learned the general two-body form of Kepler's third law — that period depends on both central mass and distance, in the correct functional relationship — from a literal handful (18) of real, noisy, historically-measured data points, and applied it correctly to a mass and distance regime it never saw in training.

This nuances the Newton/Kepler historical framing from the introduction in a way that turned out to be apt rather than merely cautionary: Newton's own generalization of gravity required evidence from *multiple* systems with varying mass (planetary orbits *and* terrestrial falling bodies), not extrapolation from a single-mass dataset. This experiment's final, successful design required the analogous thing — real training data spanning multiple central masses — before generalization to new systems became possible. The two confounds encountered along the way are not incidental noise; they are close analogues of the actual epistemic challenge historical scientists

faced in inferring a general law from necessarily incomplete data, and this paper's contribution includes surfacing exactly how easy it is for statistical unidentifiability and extrapolation gaps to masquerade as a "generalization failure" of the learning method itself.

7. Limitations

- Extremely small dataset relative to typical grokking literature (18 training bodies, 12 held-out bodies) — real solar-system data is fundamentally limited in size in a way synthetic datasets are not.
- Single physical domain (gravity/Kepler's third law) — no claim of generality to other real-world scientific relationships, which may not share its convenient log-log linearity or its relatively low measurement noise.
- **Tested and confirmed seed-sensitivity, now at n=100** (section 4.5): expanding from 2 to 100 seeds substantially changed this paper's headline claim. Successful cross-system generalization within the 60,000-step budget occurs in only 29/100 (29%) of random seeds, not reliably — the 2-seed check's 100% success rate was not representative. When generalization does succeed, held-out mechanistic exponents remain reliably close to Kepler's true values, so the *quality* of successful generalization is robust even though its *occurrence* is not. The precise grokking-delay duration and shape (originally reported as ~24,000 steps for seed=0) is confirmed not to be a fixed property of this setup — delay varies substantially across the successful seeds, and the n=20/n=100 batches used a different, more mechanical delay-detection heuristic than the manual one used for seed=0, so absolute step counts are not directly comparable across sections 4.3 and 4.5. Whether the 71% of seeds that fail to generalize within this budget would eventually succeed with more training steps was not tested.
- The n=20/n=100 seed-distribution batches ran on a PyTorch port of the original MLX training code (`orbital_train_torch.py`), needed to parallelize across CPU cores on a cloud instance rather than run single-threaded on the M1. The port was validated against the original MLX seed=0 result before being trusted at scale (final metrics and held-out exponents matched closely — see changelog, 2026-07-05), but this means sections 4.1-4.4 and the seed=0/seed=1 checks in 4.5 used MLX while the n=20/n=100 distributional results used PyTorch — a cross-framework split that, while validated, is a genuine caveat for anyone trying to exactly reproduce absolute step-count figures.
- The 12 held-out bodies are not independent tests of 12 different central masses — they cluster into only 3 actual gravitational systems (Jupiter, Saturn, Neptune), so the true "number of independent generalization tests" this result rests on is closer to 3 than 12.
- The finite-difference exponent extraction (section 4.4) estimates local slopes only in the immediate neighborhood of each existing data point (h=1e-3) — it does not verify the model behaves like a global power law far from any observed body, only that it locally looks like one near the specific points this paper cares about.
- The dataset-design process itself (identifying and fixing two confounds) was iterative and informed by seeing each result before deciding the next fix — this is normal, disclosed exploratory research practice (see changelog for the full, undisguised sequence), but it does mean the final dataset was not specified in

advance of seeing any results, unlike a pre-registered design.

8. Conclusion

Grokking — delayed generalization from memorization to a correct general rule — is not an artifact confined to synthetic, noise-free algorithmic tasks. Trained on a literal handful of real, imprecisely-measured astronomical data points, a tiny transformer both fit its training data and, roughly 18,000-24,000 steps later, correctly generalized Kepler's third law to three gravitational systems it never saw during training. Testing this at scale (100 independent seeds) shows that success is not guaranteed by the setup — only 29% of seeds generalize within this step budget — but when it happens, it reliably reflects genuine recovery of the correct physical law rather than a coincidence of a particular run. But this result did not appear on the first, most natural attempt: it required recognizing and fixing two genuine identifiability confounds in the training data — a missing distance range, then a missing mass range — each of which alone would have produced a misleadingly clean-looking negative result. The central lesson is less about the learning algorithm and more about the data: grokking's delayed-generalization behavior transfers to real data, but real data's success or failure at supporting generalization at all is much more sensitive to what, specifically, is and isn't represented in training than synthetic benchmarks — which are constructed to avoid this problem by design — ever reveal.

References

- Power, A., Burda, Y., Edwards, H., Babuschkin, I., & Sutskever, I. (2022). Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets. arXiv:2201.02177.
- Lee, J., Kang, B. G., Kim, K., & Lee, K. M. (2024). Grokfast: Accelerated Grokking by Amplifying Slow Gradients. arXiv:2405.20233.
- Thilak, V., Littwin, E., Zhai, S., Saremi, O., Paiss, R., & Susskind, J. (2022). The Slingshot Mechanism: An Empirical Study of Adaptive Optimizers and the Grokking Phenomenon. arXiv:2206.04817.

Log files referenced in this draft: - Phase 0 (modular arithmetic): `baseline_log.csv`, `baseline_wd01_log.csv`, `grokfast_log.csv`, `grokfast_30k_log.csv`, `baseline_wd03_log.csv`, `grokfast_wd03_log.csv`, `baseline_wd005_log.csv`, `grokfast_wd005_log.csv`. - Phase 1/2 (real orbital data): `orbital_baseline_log.csv` (8-body/1-mass design), `orbital_v2_log.csv` (11-body/3-mass design), `orbital_v3_log.csv` (18-body/5-mass design, 20k steps, under-converged), `orbital_v3_long_log.csv` (18-body/5-mass design, 60k steps, final result). - Phase 3 (mechanistic exponents): `orbital_v3_exponents_log.csv`, `orbital_v3_exponents_exponents.csv`. - Seed robustness check (seed=1, MLX): `orbital_v3_seed1_log.csv`, `orbital_v3_seed1_exponents.csv`. - Seed-distribution study (n=20, n=100, PyTorch port): 20-seed batch summary recorded from console output in the changelog below (not saved as a separate file); 100-seed batch summary in `orbital_batch_100.csv`. Porting/validation code: `orbital_train_torch.py`, `orbital_batch.py`.

Changelog (living doc — update every time a run finishes)

- 2026-07-04: Phase 0 harness fixed (MLX array-indexing bug in `batches()`). Baseline (wd=1.0) grokked at step 1300, confirming harness works. Found post-grok slingshot oscillation, matched to Thilak et al. 2022. wd=0.1 run did not grok in 10k steps. Grokfast 5k-step run did not clearly beat baseline.
- 2026-07-04: Ran matched Grokfast comparison (same seed=0, same 30,000-step budget as baseline). Result: Grokfast grokked at step 1900, baseline at step 1300 — Grokfast was slower, not faster.
- 2026-07-04: Ran wd=0.3 ablation to test whether Grokfast's underperformance was just a "no headroom" artifact of wd=1.0 grokking too fast already. Result: baseline grokked at step 2600, Grokfast at step 5100 — the relative gap *widened* (96% slower vs. 46% slower at wd=1.0), ruling out the headroom hypothesis.
- 2026-07-04: Ran bonus ablation at the Grokfast authors' own recommended low weight decay (wd=0.05, 20k steps). Baseline grokked at step 19,400; Grokfast never grokked in the full budget, stuck near random test accuracy throughout. Verified our Grokfast implementation against the official reference code line-by-line — matches exactly, ruling out a bug. **Final conclusion: across three weight-decay settings, Grokfast reproducibly underperformed the plain baseline in this setup, getting worse (not better) as weight decay decreased toward the authors' recommended range.** Phase 0 is complete and fully written up. **Next:** Phase 1 — real orbital data. Planetary and moon data pulled (NASA/NSSDC via astronomynotes.com compiled tables) into `orbital_data.py`; adapted training script `orbital_train.py` written (regression head, MSE loss on log10(period), log-scaled mass/distance inputs). Design decisions made: (1) central mass = mass of the body being orbited (Sun for planets, host planet for moons) per Kepler's third law, meaning central mass is constant across all 8 Phase-1 planets and only starts varying in Phase 2 (moons) — this is the real test of whether the model learned the general law or just curve-fit distance; (2) train on all 8 planets, no Phase-1 holdout (6 training points would be too thin to draw any conclusion from a 2-point test set) — Phase 2's 9 moons carry the real generalization claim; (3) log10-space loss/inputs throughout, since raw values span orders of magnitude (88 to 60,000+ day periods) and Kepler's law is linear in log-log space anyway. Uranus moon data not yet pulled — flagged as a placeholder in `orbital_data.py`, not fabricated.
- 2026-07-04: Ran Phase 1/2 (`orbital_train.py`, wd=0.1, 20k steps, seed=0). Phase 1 (8 planets) fit essentially perfectly (train MSE ~0.0001-0.0005, all predicted periods within ~10% of true). Phase 2 (9 held-out moons) did not converge and showed systematic underestimation, worsening with distance from the training scale (1.3x-14.1x error). Root cause identified: the Sun's mass is constant across all Phase 1 training rows, so the mass-dependence term of Kepler's third law is unidentifiable from this data regardless of training procedure — a data-design limitation, not a grokking/architecture failure (see section 4.3, 6). A second, independent confound was also found: every Phase-2 moon distance falls below the entire Phase-1 planetary distance range with zero overlap, so the failure also conflates "didn't learn mass-dependence" with "can't extrapolate past the trained distance range."
- 2026-07-05: **Decision (with user sign-off): redesign rather than report the**

above as final. Added Earth's Moon and Mars' two moons (Phobos, Deimos) to Phase 1 training — real bodies with central masses other than the Sun (Earth, Mars), chosen so Earth's Moon's distance overlaps the Phase-2 test range. Phase 1 training is now 11 real bodies across 3 central masses (was 8 bodies, 1 central mass). Phase 2 unchanged — Jupiter/Saturn/Neptune's moons remain fully held out. Updated `orbital_data.py` with the new bodies and a design note explaining why. **Next:** re-run `orbital_train.py` with the updated dataset. If the model now correctly extrapolates mass-dependence (with Earth's Moon providing distance-range overlap), that's a real positive transfer result; if it still fails, the identifiability confound is ruled out and the failure would be more directly attributable to the model/ training itself. Phase 3 (mechanistic exponent extraction) remains pending until this re-run's outcome is known — its usefulness depends on whether the mass term was actually learnable in the updated design.

- 2026-07-05: Ran the re-run (`orbital_v2_log.csv`). Train MSE converged to near-zero across all 11 bodies, but held-out MSE still did not converge (oscillated ~0.6 for the full run), and every held-out prediction collapsed toward ~27-29 days regardless of true period. Diagnosed the cause: the same "gap" confound recurred, but in mass-space — training central masses (log10 kg: Mars 23.81, Earth 24.78, Sun 30.30) left a 5.5-order-of-magnitude gap, and every held-out central mass (Neptune 26.01, Saturn 26.75, Jupiter 27.28) fell inside it, unrepresented in training.
- 2026-07-05: **Second redesign (with user sign-off, scoped down from an initial "every object in the solar system" ask to "all planets + major moons only, no minor moons/asteroids").** Pulled and verified real NASA/NSSDC/JPL data (semi-major axis + orbital period only — central mass, not the orbiting body's own mass, is the only feature that matters) for: Uranus's 5 major moons (Miranda, Ariel, Umbriel, Titania, Oberon), Pluto (as a 9th Sun-orbiting planet), Charon (orbits Pluto), and Saturn's remaining 3 major moons (Mimas, Tethys, Dione). Design split: bodies whose central mass fills the training gap (Uranus's moons, Charon) went to **Phase 1 training**, since Uranus and Pluto are not Phase-2 test systems; Saturn's new moons went to **Phase 2** since Saturn already is a held-out test system and adding its moons to training would leak test-system data. Phase 1 is now 19 real bodies across 5 central masses (Sun, Earth, Mars, Uranus, Pluto); Phase 2 is now 12 bodies across Jupiter, Saturn, Neptune. Updated `orbital_data.py` (DESIGN NOTE 2) accordingly. **Next:** re-run `orbital_train.py` with the expanded dataset and check whether held-out MSE finally converges now that the model has training signal spanning the full mass range it will be tested on.
- 2026-07-05: Ran the expanded-dataset re-run at 20k steps (`orbital_v3_log.csv`). Held-out predictions no longer collapsed to a constant (real progress), but a new confound appeared: train MSE only reached ~0.01 (vs. ~0.0001-0.0005 previously) and was still visibly oscillating — training itself hadn't converged on the harder, more diverse 18-point dataset within the same step budget that sufficed for the easier 8-11 point versions.
- 2026-07-05: Re-ran at 60,000 steps, same seed/hyperparameters (`orbital_v3_long_log.csv`). Train MSE converged to ~0.0003-0.0004; held-out MSE converged to a closely matched ~0.0003-0.0008. All 12 held-out moons (Jupiter, Saturn, Neptune) predicted within ~12% of true period, most within 5% — comparable to the

training fit's own accuracy. **This is the paper's headline positive result: a tiny transformer learned the general (mass- and distance-dependent) form of Kepler's third law from 18 real, noisy measurements and correctly generalized it to 3 unseen gravitational systems.** Both confounds that produced the two earlier negative-looking results (distance-range gap, then mass-range gap) were real, diagnosed, and fixed with genuine additional real-world data rather than by reframing the failures. Updated section 4.3 (full results table), section 5 (results summary), and section 6 (discussion) to reflect the final outcome. **Next:** Phase 3 (extract the model's implicitly learned mass/distance exponents and compare to Kepler's third law's true values) is now meaningful to run, since this result confirms the relationship was actually learned rather than curve-fit to a single system.

- 2026-07-05: Added `--extract_exponents` to `orbital_train.py` (finite- difference local slope estimation, no retraining/model-saving needed — runs immediately after training completes, in the same process). Ran on the same 60k-step config as the successful `orbital_v3_long` run (`orbital_v3_exponents_log.csv`, `orbital_v3_exponents_exponents.csv`). Held-out (12-body) mass exponent: -0.4481 ± 0.0259 (true: -0.5). Held-out distance exponent: 1.4166 ± 0.0439 (true: 1.5). Both tightly clustered and close to Kepler's true values across all 3 held-out systems — mechanistic confirmation that section 4.3's generalization result reflects the model actually implementing something close to the true two-body law, not a coincidence of outputs. Training-set exponents are noisier, especially near Neptune/Pluto, plausibly because 9 of 18 training points share the Sun's exact mass, leaving little local mass-direction variation to estimate a precise slope from near those specific points — not evidence against the finding, just a finite-difference estimation-precision artifact at the edge of the training distribution. Updated section 4.4 (full write-up), section 5 (results table), and the abstract to reflect this. **Phase 3 is complete.**
- 2026-07-05: Ran a second-seed robustness check (seed=1, identical final design, 60k steps, `--extract_exponents`) to test the seed-sensitivity limitation flagged in section 7. Result: the headline finding replicates — successful cross-system generalization (held-out errors mostly <5%, worst case -29.5% on Iapetus, the most extreme distance-outlier moon in both seeds) and mechanistic confirmation (held-out exponents -0.4698 ± 0.0138 mass, 1.3982 ± 0.0691 distance, vs. true -0.5/1.5, again tighter than the noisier train-set estimates). However, the clean single-cliff grokking shape from seed=0 does NOT replicate — this seed shows a noisier, longer, multi-episode approach (train doesn't settle until ~step 37,600; a large train+held-out relapse occurs at steps 45,000-46,600 before final recovery). Updated section 4.3 (new 4.5 subsection), abstract, results table, and limitations to report this honestly: the qualitative finding (delay exists; generalization eventually succeeds; exponents are recovered) is robust across 2 seeds, but the precise ~24,000-step figure is not a fixed property of this setup.
- 2026-07-05: Ported the training code to PyTorch (`orbital_train_torch.py`) to allow parallel multi-seed batches on cloud CPU instances (MLX is Apple-Silicon-only). Validated against the original MLX seed=0 result before trusting it at scale: final train MSE 0.00245 (MLX-comparable), final held-out MSE 0.00401, held-out mass exponent -0.4515 ± 0.0249 (MLX: -0.4481 ± 0.0259), held-out distance exponent 1.3978 ± 0.0906 (MLX: 1.4166 ± 0.0439) — close match on

the metrics that matter, though the grokking-delay heuristic gives a different absolute step count than the manual MLX inspection (expected, since the two use different threshold-crossing definitions). Wrote `orbital_batch.py` to run many seeds in parallel via multiprocessing. Provisioned an AWS c7i.8xlarge instance (eu-west-1) via CLI to run batches beyond what the M1 can do in parallel.

- 2026-07-05: Ran a 20-seed timing/sanity batch on the AWS instance (seeds 0-19, `orbital_batch.py`, 60k steps each). Result: only 8/20 (40%) succeeded (final held-out MSE < 0.01) — a materially lower success rate than the 2/2 (100%) the seed=0/seed=1 spot check had suggested. Held-out exponents among successes: mass -0.4436 ± 0.0308, distance 1.3638 ± 0.0316, consistent with the smaller checks. This result prompted scaling to a larger batch to get a more precise success-rate estimate.
- 2026-07-05: Ran a 100-seed batch on the same AWS instance (seeds 0-99, `orbital_batch_100.csv`, 60k steps each, ~73 min total wall time). Result: 29/100 (29%) succeeded — narrower than but consistent in direction with the 20-seed check (40%), confirming successful generalization is genuinely a minority outcome at this step budget, not a fluke of a small sample. Held-out exponents among the 29 successes: mass -0.4320 ± 0.0426, distance 1.3692 ± 0.0486 (true: -0.5, 1.5) — consistent with every smaller check. Grokking delay among successes (mechanical threshold heuristic): mean 2772, stdev 4040, min -200 (heuristic artifact, not a real result), max 13200. Updated the abstract, introduction (added contribution 4), method (3.3, hardware/tooling note), section 4.5 (added the scaling-to-many-seeds subsection), section 5 (results table), section 7 (limitations, rewrote seed-sensitivity bullet and added a framework-split caveat), and section 8 (conclusion) to reflect this as the paper's authoritative seed-robustness finding, superseding the 2-seed and 20-seed checks. Terminated the AWS instance after downloading `orbital_batch_100.csv`. **This paper's full experimental arc (Phase 0 sanity check → Phase 1/2 real-data generalization, including two diagnosed-and-fixed confounds → Phase 3 mechanistic confirmation → seed-robustness check, scaled from 1 to 2 to 20 to 100 seeds) is now finished and fully written up.**