ORDER, EXPOSURE, OR SUPPORT? DISENTANGLING CURRICULUM EFFECTS ON GROKKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Grokking is when a network generalizes abruptly, long after it has fit its training set. It is sensitive to how the training data are presented, but "presentation" quietly bundles three things. A curriculum that shows hard examples first changes the *order* of examples, changes how often each is seen (*exposure*), and keeps the optimizer on a narrow slice of the data for a while (*support*). We separate them with two controlled experiments on small algorithmic tasks. First, we fix per-example exposure over the whole run and vary only the order. Sorting by difficulty, in either direction, then prevents grokking outright on four of six tasks, with validation accuracy at chance for the whole run, and delays it 6–54× on the other two. So exposure counts are not the cause. Second, on one task from each model scale, we make every epoch contain the full dataset and vary only the order within each epoch. The orders-of-magnitude effect disappears: sorted and shuffled epochs reach the transition within one evaluation interval. So fine-grained order is not the cause either. Across all our schemes, the transition tracks one quantity: how long the optimizer is confined to an unrepresentative slice of the difficulty range. Sorting within 9-step epochs changes nothing, sorting within 156-step epochs costs at most 1.5×, growing pools that confine early cost 16–59×, and a sort that confines for the whole run blocks grokking. This places curriculum effects inside a known property of grokking, its steep dependence on the fraction of data used for training: a curriculum lowers the effective training fraction for as long as it confines the support. We state this as the leading explanation, give the experiments that would falsify it, and release all 505 runs with per-step trajectories.

1 INTRODUCTION

Grokking (Power et al., 2022) is the phenomenon where a model fits its training set long before held-out accuracy abruptly improves. It has become a controlled probe for memorization, feature learning, weight decay and capacity (Nanda et al., 2023; Liu et al., 2023; Barak et al., 2022), and it is known to depend on how data are presented. We ask which part of presentation actually moves the transition.

Curriculum learning traditionally presents examples from easy to hard (Bengio et al., 2009). The growing-pool curricula common in practice start from the easiest (or hardest) 10% of the data and widen to the full set over training. Such a schedule changes three things at once:

- **Order**: the sequence in which examples appear.
- **Exposure**: under a fixed step budget, examples admitted early are seen far more often.
- **Support**: for a stretch of training, the optimizer only sees a narrow, unrepresentative slice of the data.

A growing-pool experiment cannot tell these apart, so its effects on grokking are hard to interpret. On mod-97 addition, for example, such curricula delay grokking 51–59× (§4.1), but the delay could come from any of the three.

We separate them one at a time, on six algorithmic tasks where grokking is clean and runs are cheap:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Exposure is not the cause (§4.2). We fix a multiset in which every example appears equally often over the run, and vary only the sequence. Sorting by difficulty, in either direction, still prevents grokking on four of six tasks and delays it 6–54$\times$ on the rest.

Fine-grained order is not the cause (§4.3). We make every epoch contain the whole dataset once, and sort only within epochs. The orders-of-magnitude effect disappears.

Confinement to a narrow support is the remaining explanation (§4.4). Across every scheme we ran, the transition is ordered by how long the optimizer is kept on an unrepresentative slice, and the evidence that varies this most directly (dataset size, the length of a hard-first phase) moves the transition the way a support account predicts.

Why confinement matters for grokking. Grokking time rises steeply as the fraction of the data used for training shrinks, and below a task-dependent fraction a model memorizes without generalizing within a practical budget (Power et al., 2022; Liu et al., 2023). A curriculum that confines the optimizer to a slice of the data lowers the effective training fraction for as long as the confinement lasts. The support account therefore places curriculum effects inside a known property of grokking rather than beside it: order matters only through how it restricts the data the optimizer currently sees.

The practical reading is simple: a curriculum is risky to the extent that it keeps the model on a narrow slice of the data for a sustained stretch, and harmless once each pass over the data is representative.

Scope. All controlled experiments use small transformers on algorithmic tasks at the data fractions where grokking is usually studied. At much sparser data, a fixed ordering can itself carry useful signal (LeDoux, 2026); our claim concerns difficulty curricula in the standard regime, where we find that order acts through the support it creates.

Contributions. (1) An exposure-matched control showing that curricula prevent or delay grokking even when every example is seen equally often. (2) A prefix-matched control showing that once each epoch is representative, within-epoch order does not matter. (3) Converging evidence that confinement to an unrepresentative support is the operative variable, including a graded relation between how long each scheme confines the optimizer and how much it delays grokking, and a link to the known dependence of grokking on training fraction. (4) Explicit predictions that would falsify the support account. (5) All 505 runs released with per-step trajectories, so every number recomputes without a GPU.

2 Related Work

Grokking. Work since Power et al. (2022) has studied progress measures and circuits (Nanda et al., 2023; Varma et al., 2023), weight-norm and loss-geometry accounts (Liu et al., 2023; Thilak et al., 2022), hidden progress in sparse parity (Barak et al., 2022), the lazy-to-rich transition (Kumar et al., 2024), implicit-bias dichotomies (Lyu et al., 2024), numerical stability (Prieto et al., 2025) and grokking in multi-hop QA (Abramov et al., 2025). These vary the data distribution, the model or the optimizer. We hold all three fixed and vary only how a fixed dataset is presented.

Curriculum and data ordering. Curriculum learning (Bengio et al., 2009; Soviany et al., 2022) is usually studied as a speed or accuracy heuristic. On standard benchmarks, Wu et al. (2021) found that most of the benefit of curricula comes from the dynamically growing dataset size rather than from the ordering itself; our results show that the same variable governs grokking, and separate exposure from support at a fixed dataset, which a growing dataset cannot. Recent work compares easy-to-hard and reverse curricula across difficulty metrics for LLM mathematical reasoning and finds no universally best direction (Jia et al., 2026). We study the timing of a generalization transition, and add the prefix-level control that separates order from support.

Ordering as a signal. LeDoux (2026) show that a fixed, structured ordering, repeated every epoch, can make modular addition generalize from very sparse data (0.3% of all pairs) where shuffled training does not, and that at a data density closer to ours (26.5%) fixed orderings are only about 20% faster than shuffling. Our findings are consistent with that picture and complementary to it. In the data regime where grokking is usually studied (here 50% of pairs for modular arithmetic), sorting within an epoch leaves the transition unchanged or changes it by at most 1.5$\times$, while confining the support moves it by orders of magnitude. Both effects can hold at once: order can carry information when data are scarce, and support restriction can block generalization when it lasts.

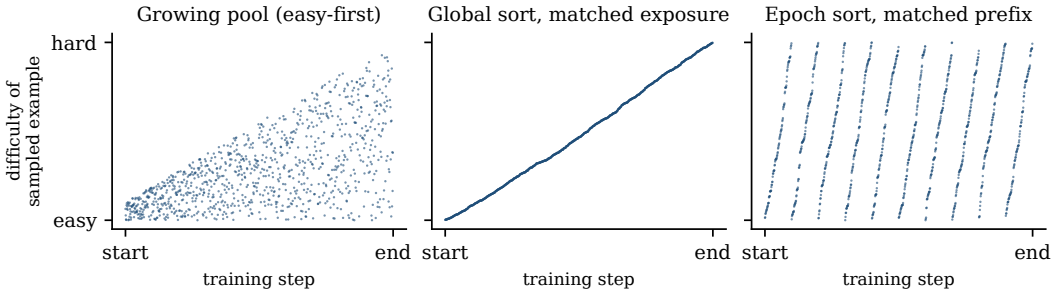


Figure 1: **The three presentation schemes**, illustrated with easy-first ordering. Each dot is one sampled example. A growing pool changes order, exposure and support together. A global sort equalizes exposure over the run but keeps the support narrow at every point in training. An epoch sort equalizes exposure at every epoch boundary and covers the full difficulty range in every epoch.

Grokking and training-set size. Grokking time diverges as the training set shrinks toward a critical size (Power et al., 2022; Liu et al., 2022), partial coverage of the input distribution has been argued to drive delayed generalization (Carvalho et al., 2025), and a recent multi-seed study finds the grokking transition gated by training-set coverage (Ootani, 2026). The support account connects curriculum effects to this line of work: a curriculum shrinks the effective training set for as long as it confines the support.

Non-stationary training. A globally sorted stream is non-stationary, and continual learning shows that such streams cause forgetting (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017). Our question is different: not whether earlier examples are forgotten, but whether a generalizing solution forms at all while the support is narrow. Forgetting and support restriction are two views of the same manipulation, and §5 discusses what our data can and cannot separate.

Forecasting grokking. Notsawo et al. (2023) predict *whether* grokking occurs from early loss-landscape features. Our spectral diagnostic predicts *when*, from weight singular values alone.

3 Setup

Tasks and models. Six algorithmic tasks at two scales. Group A (2-layer transformer, $d = 128$): addition and multiplication modulo 97, and S_5 permutation composition; 9,409–14,400 pairs, 50/50 train/validation. Group B (4-layer transformer, $d = 256$): 4-digit addition, length-8 sorting, and GCD over $[1, 1000]$; 50,000 examples, 80/20. Both use AdamW (learning rate 10^{-3} / 3×10^{-4}, weight decay 1.0 / 0.1, batch 512 / 256) for 100K / 50K steps. Difficulty is task-specific: $\max(a, b)/97$ for modular arithmetic, number of non-fixed points for S_5, carries for addition, inversions for sorting, and Euclidean steps for GCD (Appendix B).

Transition time. t^* is the first step at which a run's validation accuracy reaches 0.9, a threshold fixed before any analysis and never tuned per scheme. "Never" means not within the step budget. Validation is evaluated every 250 steps, and tables report the mean over three seeds.

Presentation schemes. All schemes use the same data, model, optimizer and step budget.

- **Growing pool** (the usual curriculum): at training fraction f, sample from the easiest (or hardest) $0.1 + 0.9f$ quantile of difficulty. Order, exposure and support all change together.
- **Global sort, matched exposure**: tile the training set to the full step budget so every example appears equally often, then sort the whole multiset once by difficulty (ascending or descending, ties broken at random) and draw sequential batches. The shuffled control applies a random permutation to the same multiset.
- **Epoch sort, matched prefix**: each epoch is one full pass over the dataset, sorted within the epoch (or shuffled). Exposure is equal at every epoch boundary.

4 RESULTS

4.1 GROWING-POOL CURRICULA SHIFT GROKKING, BUT CONFOUND THREE FACTORS

On mod-97 addition, schedules that sample from the whole dataset grok by about step 1,100, whether uniform, entropy-sorted or alternating between easy and hard halves. Growing-pool curricula are far slower (Table 1): hard-first reaches $t^* = 55{,}333$ and easy-first $64{,}000$, a 51–59$\times$ delay, with anti-curriculum (hard for the first half, easy for the second) in between. The same ordering holds on the other tasks (Appendix C), and every curriculum eventually exceeds 99% validation accuracy. Note that the two directions behave alike: difficulty-restricted sampling delays grokking whether it starts from the easy end or the hard end.

Table 1: Growing-pool curricula on mod-97 addition: t^* (mean over three seeds). These schedules change order, exposure and support together.

Schedule	t^*	vs. uniform
Uniform	1,083	1$\times$
Entropy-sorted, alternating	$\approx$1,100	$\approx$1$\times$
Anti-curriculum (hard then easy)	24,500	23$\times$
Hard-first	55,333	51$\times$
Easy-first	64,000	59$\times$

4.2 MATCHED EXPOSURE: EXPOSURE COUNTS ARE NOT THE CAUSE

Table 2 holds per-example exposure identical across arms and varies only the sequence. A random shuffle groks fastest on every task. Sorting by difficulty, in either direction, prevents grokking on the two modular tasks, S_5 and 4-digit addition: validation accuracy stays at chance for the entire run while the model continues to fit the examples it is being shown. "Never" is censored at the step budget, 16–86$\times$ the shuffled t^*; we do not claim these runs would never grok with unlimited training. On GCD and sorting it delays grokking 6–54$\times$. Because every example is seen equally often in all three arms, the delay cannot come from exposure counts.

Table 2: Matched exposure: t^* when every example is seen equally often over the run and only the order differs. "Never": no grokking within the budget.

Task	Shuffled	Sorted easy$\rightarrow$hard	Sorted hard$\rightarrow$easy
mod-97 addition	1,167	never	never
mod-97 multiplication	1,167	never	never
S_5 composition	6,167	never	never
4-digit addition	667	never	never
GCD	500	12,250	27,250
sort-8	750	4,417	4,500

A global sort equalizes exposure over the run but not over every prefix of it. For long stretches, the sorted arms have seen only one end of the difficulty range. So this experiment rules out exposure counts, but by itself does not separate order from sustained support restriction.

4.3 MATCHED PREFIX: FINE-GRAINED ORDER IS NOT THE CAUSE

If order itself were the lever, sorting within each epoch should still matter. It does not (Table 3). When every epoch contains the full dataset, the three within-epoch orders on mod-97 addition reach the threshold at the same evaluation step, and on GCD they fall within a few hundred steps of each other. Contrast Table 2: the same difficulty sort, applied globally rather than within epochs, prevents grokking on mod-97 addition entirely. The difference between the two experiments is not the order within any stretch of data, which is sorted in both, but whether each stretch covers the whole difficulty range.

Table 3: Matched prefix: t^* when every epoch contains the full dataset once and only the within-epoch order varies. At our evaluation interval (250 steps), the orders are indistinguishable on mod-97 addition.

Task	Shuffled per epoch	Easy→hard per epoch	Hard→easy per epoch
mod-97 addition	1,167	1,167	1,167
GCD	500	583	750

The prefix-matched control covers one task from each group. The exposure-matched result of §4.2 covers all six, and the conclusion rests on the two together.

4.4 Confinement to a narrow support

With exposure counts and fine-grained order ruled out, the remaining difference between schemes is how long the optimizer is confined to an unrepresentative slice of the difficulty range. Four pieces of evidence point the same way.

How long each scheme confines the optimizer. The schemes differ in how many steps pass before the optimizer has again seen the whole difficulty range (Table 4). With 4,704 training pairs and batch 512, an epoch on mod-97 addition lasts about 9 steps, so an epoch sort revisits every difficulty level every 9 steps; on GCD (40,000 training examples, batch 256) an epoch lasts about 156 steps. A global sort on mod-97 addition spreads 97 difficulty levels over 100K steps, so the optimizer spends on average about 1,000 consecutive steps on one level and sees the full range only once, across the whole run. A growing pool starts from 10% of the training set and reaches all of it only at the end.

Table 4: Confinement and delay. Confinement is how long the optimizer trains before it has again seen the full difficulty range, derived from each scheme's construction. Delay is t^* relative to shuffled presentation, over both directions.

	Confinement		Delay vs. shuffled	
Scheme	mod-97 addition	GCD	mod-97 addition	GCD
Epoch sort	≈9 steps	≈156 steps	1.0×	1.2–1.5×
Growing pool	widens over the run	widens over the run	51–59×	16–31×
Global sort	whole run	whole run	never groks	25–55×

The schemes are ordered by confinement. Table 4 and Figure 2 line up every scheme by how long its support stays narrow. Epoch sorting never confines the optimizer, and it groks as fast as uniform sampling. Growing-pool curricula confine it early and then widen to the full dataset, and they grok late but do grok. The global sort confines it for the whole run, and it never groks. On GCD, where every scheme eventually groks, the same order holds in both directions: epoch sort at 583–750 steps, growing pool at about 8,000 (easy-first) and 15,600 (hard-first), and global sort at 12,250 and 27,250 (Appendix C).

The delay is graded, not all-or-none. Table 4 orders the schemes the same way on both tasks. It also shows a residual effect exactly where the account predicts one: epoch sorting leaves no delay on mod-97 addition, where an epoch lasts 9 steps, but a small one (1.2–1.5×) on GCD, where an epoch lasts 156 steps.

Grokking arrives once the support is broad enough. Under a growing pool the support widens on a known schedule ($0.1 + 0.9f$ of the training set at training fraction f), so the mean t^* tells us how much of the training set the pool held when the runs grokked. On mod-97 addition that is 60% (hard-first) and 68% (easy-first); on GCD, 38% and 24%. Runs do not need the full training set to grok, only enough of it, and how much is enough depends on the task, as the training-fraction threshold does for grokking on a fixed dataset (Power et al., 2022). Under a global sort, the support within any stretch shorter than the whole run stays narrow, which is why it never groks on mod-97 addition.

Neither direction is consistently better. Easy-first and hard-first delay grokking by similar amounts, and which one is slower flips across tasks: on mod-97 addition the growing-pool de-

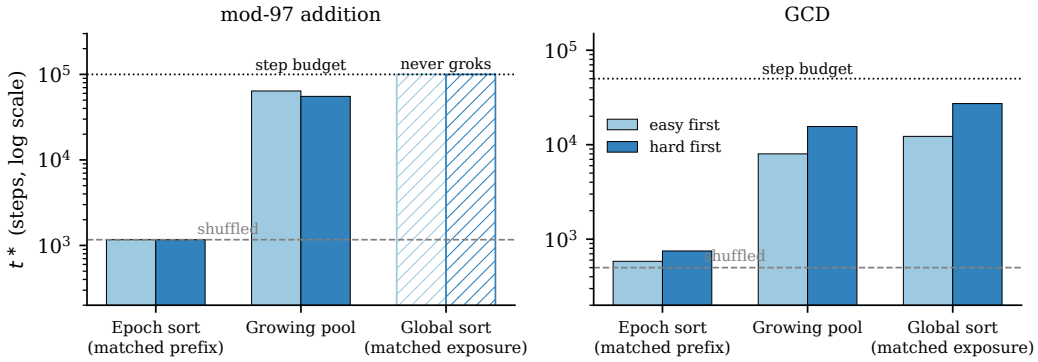


Figure 2: **Transition time follows how long the support stays narrow.** t^* (log scale) under each presentation scheme, in both directions. Epoch sorting never narrows the support and groks near the shuffled reference; growing pools narrow it early; a global sort narrows it throughout and, on mod-97 addition, never groks within the budget.

lays are 59× (easy-first) and 51× (hard-first); on GCD the global-sort delays are 25× and 55×; on sorting they are equal; and both global-sort directions prevent grokking on the same four tasks. An account in which learning easy structure first helps would predict a consistent asymmetry. A support account predicts none, since either end of the range is unrepresentative.

Larger datasets shrink the delay. Growing the training set widens every slice in absolute terms: the first 10% of a larger dataset contains more distinct examples. Consistent with a support account, in a separate data-scale sweep on GCD the growing-pool delay falls monotonically with dataset size, from 84× at 5K examples to 52× at 10K (Appendix E).

A short narrow phase costs little. Varying how long an anti-curriculum stays in its hard-first phase, on GCD a brief hard phase followed by easy examples (25% of training) is the fastest schedule we tested ($t^* = 5{,}250$), faster than easy-first throughout (8,000) or hard-first throughout (15,583) (Appendix F). What hurts is sustained confinement, not the presence of a narrow phase.

Taken together, these results make confinement duration the leading explanation. We have not varied block length at fixed exposure in a single experiment, so we state it as the explanation our data support rather than as a proven cause. A global sort could also change gradient autocorrelation or the number of distribution shifts, and those move together with confinement in our schemes.

Optimizer methods act on a different axis. Two methods designed to change grokking do not undo the curriculum effect. GrokFast (Lee et al., 2024) speeds up easy-first grokking on GCD by 1.7× (8,000 to 4,833 steps) but leaves the ordering uniform < easy-first < hard-first intact. SAM (Foret et al., 2021) suppresses the sharp hard-first transition on mod-97 addition (largest single-step jump 0.39 to 0.04) while leaving t^* unchanged or slightly later (Appendix G). Curriculum support and loss-geometry methods are composable, not substitutes.

4.5 Predictions that would falsify the support account

The support account is supported by the ordering of every scheme we ran, but no single experiment here varies confinement alone. It therefore makes three predictions that a direct test could refute, each holding exposure fixed. *Block length:* sorting within blocks of B epochs should delay grokking monotonically in B, interpolating between epoch sorting ($B = 1$, no delay) and the global sort (no grokking). *Slice width at fixed dwell time:* visiting difficulty slices in random order, so that there is no easy-to-hard direction, should delay grokking more for narrower slices and longer dwell. *Replay:* mixing a small fraction of uniformly sampled examples into a globally sorted stream should restore grokking, since it restores representative support without changing the order of the rest. If block length had no effect, or replay failed to restore grokking, the account would be wrong. All three use the released training code and change only the sampler.

An early spectral diagnostic. As a secondary result, a two-feature regression on weight singular values at the first checkpoint forecasts $\log t^*$ without validation signal from the run, with fully nested

leave-one-cell-out R^2 of 0.66–0.81 across regimes (Appendix H). Because curricula change the data seen from the first step, part of this signal may identify the curriculum rather than anticipate the transition, and we do not rely on it for any claim.

4.6 SCOPE: PRETRAINED TRANSFORMERS

We also ran growing-pool curricula while fine-tuning pretrained GPT-2 on multi-hop question answering (Appendix J). These runs are not exposure-controlled, and training loss is not near zero at the transition, so they show a curriculum-induced sharp validation transition rather than classical grokking. We report them as a map of where such a transition appears.

On 2WikiMultiHopQA (Ho et al., 2020), uniform and easy-first schedules reach validation accuracy $\approx$0.85 at the first evaluation. Hard-first schedules instead sit near 0.33 and then jump to $\approx$0.84 within one 500-step window (steps 2,750–3,250 in all three seeds). Table 8 in Appendix J summarizes what moves the jump. It lands in the same window at 125M, 355M and 1B parameters, across two model families. It depends on the difficulty signal: ordering by supporting-hop count alone reproduces it with zero across-seed variance, while ordering by GPT-2 perplexity or at random removes it. It needs pretraining: from-scratch models from 2 to 12 layers never leave chance. And it is specific to this dataset: HotpotQA (Yang et al., 2018) and GSM8K (Cobbe et al., 2021) show no transition, and MuSiQue (Trivedi et al., 2022) shows the low-accuracy phase but recovers gradually. Breaking bridge-entity alignment between supporting paragraphs does not remove the jump, which leaves the explicit knowledge-graph evidence triples in the 2WikiMultiHopQA prompt as the most plausible dataset-specific trigger. Because hop count is integer-valued, a new hop stratum enters the growing pool at a fixed step, so the jump may coincide with a stratum becoming available; an exposure-controlled version of this experiment is the natural next step.

5 DISCUSSION AND LIMITATIONS

What a curriculum does to grokking. Holding exposure fixed, a global sort blocks grokking. Balancing every epoch removes the effect. Across schemes, the transition follows how long the support stays narrow, shows no consistent preference for either direction, shrinks as the dataset grows, and tolerates a brief narrow phase. The lever is how long the optimizer sees an unrepresentative slice of the data, not how often each example is seen and not the fine-grained order. Growing-pool curricula mix all three, which is why their effects on grokking have been hard to interpret.

Practical rule. A difficulty-based schedule is safe for grokking when each pass over the data is representative, and risky to the extent that it holds the model on a narrow slice for a sustained stretch. If a curriculum is wanted for other reasons, applying it within epochs rather than across training preserves its ordering without the delay.

Alternative explanations. Four alternatives fit parts of the data; none fits all of it. *Exposure counts* are identical in the matched-exposure and matched-prefix runs, which still differ by orders of magnitude. *Non-stationarity or forgetting*: an epoch-sorted stream is non-stationary too, drifting from easy to hard every 9 steps on mod-97 addition, yet it groks on schedule; what separates it from a global sort is the timescale of the drift, which is the support account's variable, not a competing one. *A particular difficulty metric*: the effect appears under five unrelated difficulty definitions across six tasks (operand size, permutation fixed points, carries, inversions, Euclidean steps) and in both directions, so it is not a property of one ordering. *Gradient autocorrelation*: consecutive batches are more alike under a global sort, and autocorrelation rises with confinement duration in all our schemes, so the two cannot be separated with these data; shuffling within narrow slices at a fixed dwell time would separate them.

Limitations. (1) The prefix-matched control covers two of six tasks, at a 250-step evaluation interval; it resolves differences larger than one interval, not smaller. (2) The support account is supported by converging evidence across schemes rather than a single experiment that varies confinement length at fixed exposure; such an experiment is the natural next step. (3) All controlled results are on small transformers trained from scratch on algorithmic tasks, with three seeds per configuration. (4) The spectral diagnostic rests on 18 runs per regime and does not transfer across regimes as a single predictor. (5) We use three seeds per configuration. Grokking outcomes near a threshold can flip for a minority of seeds under small numerical perturbations (Ootani, 2026); the effects we report are

larger in kind: on the four tasks where a global sort prevents grokking, no sorted run grokked and every shuffled run did. (6) Each task group uses one optimizer setting (AdamW with weight decay 1.0 or 0.1); how confinement interacts with weight decay is untested.

6 Conclusion

Curricula change grokking, but not for the reason the word "curriculum" suggests. When per-example exposure is held fixed and only the order varies, difficulty sorting still prevents grokking on four of six tasks, so how often examples are seen is not the mechanism. When every epoch covers the full difficulty range, the effect disappears, so fine-grained order is not the lever either. What remains, and what every scheme we ran is ordered by, is how long the optimizer is confined to an unrepresentative slice of the data. In effect, a curriculum lowers the effective training fraction for as long as it confines the support, and grokking is known to be highly sensitive to that fraction. A curriculum is risky to the extent that it keeps the model on such a slice for a sustained stretch, and harmless once each pass through the data is representative. The released runs let others test this account directly, for instance by varying block length at fixed exposure.

Reproducibility Statement

All code, per-run configurations and per-step training logs for all 505 runs are in the supplementary material, together with the scripts that produce every table. Every t^*, interval and R^2 in the paper recomputes from the released logs without a GPU. Hyperparameters are in Appendix B; the curriculum samplers and the exposure- and prefix-matched constructions are specified in the Setup section and Appendix B.

AI Use Statement

In this work, we used generative AI tools for prose editing, grammar and LaTeX boilerplate. We have not used generative AI tools for experimental design, writing experiment code, generating results, or selecting which results to report, and AI-generated research ideas are not applicable to this work. Every citation was checked against its source by the authors. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

References

Roman Abramov, Felix Steinbauer, and Gjergji Kasneci. Grokking in the wild: Data augmentation for real-world multi-hop reasoning with transformers. In *International Conference on Machine Learning (ICML)*, 2025.

Boaz Barak, Benjamin L. Edelman, Surbhi Goel, Sham M. Kakade, Eran Malach, and Cyril Zhang. Hidden progress in deep learning: SGD learns parities near the computational limit. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning (ICML)*, 2009.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning (ICML)*, pp. 2397–2430, 2023.

Breno W. Carvalho, Artur S. d'Avila Garcez, Luís C. Lamb, and Emílio Vital Brazil. Grokking explained: A statistical phenomenon. *arXiv preprint arXiv:2502.01774*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations (ICLR)*, 2021.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of COLING*, 2020.

Yaning Jia, Chunhui Zhang, Xingjian Diao, Xiangchi Yuan, Zhongyu Ouyang, Chiyu Ma, and Soroush Vosoughi. What makes a good curriculum? Disentangling the effects of data ordering on LLM mathematical reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 34472–34488, 2026.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Tanishq Kumar, Blake Bordelon, Samuel J. Gershman, and Cengiz Pehlevan. Grokking as the transition from lazy to rich training dynamics. In *International Conference on Learning Representations (ICLR)*, 2024.

Jordan LeDoux. The order is the message. *arXiv preprint arXiv:2603.25047*, 2026.

Jaerin Lee, Bong Gyun Kang, Kihoon Kim, and Kyoung Mu Lee. Grokfast: Accelerated grokking by amplifying slow gradients. *arXiv preprint arXiv:2405.20233*, 2024.

Ziming Liu, Ouail Kitouni, Niklas Nolte, Eric J. Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Ziming Liu, Eric J. Michaud, and Max Tegmark. Omnigrok: Grokking beyond algorithmic data. In *International Conference on Learning Representations (ICLR)*, 2023.

Kaifeng Lyu, Jikai Jin, Zhiyuan Li, Simon S. Du, Jason D. Lee, and Wei Hu. Dichotomy of early and late phase implicit biases can provably induce grokking. In *International Conference on Learning Representations (ICLR)*, 2024.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, 1989.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *International Conference on Learning Representations (ICLR)*, 2023.

Pascal Jr. Tikeng Notsawo, Hattie Zhou, Mohammad Pezeshki, Irina Rish, and Guillaume Dumas. Predicting grokking long before it happens: A look into the loss landscape of models which grok. *arXiv preprint arXiv:2306.13253*, 2023.

Yoshiyuki Ootani. Grokking is conditional and fragile: A fully-tractable, multi-seed study at 12K parameters. *arXiv preprint arXiv:2607.05104*, 2026.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Lucas Prieto, Melih Barsbey, Pedro A. M. Mediano, and Tolga Birdal. Grokking at the edge of numerical stability. In *International Conference on Learning Representations (ICLR)*, 2025.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6):1526–1565, 2022.

Vimal Thilak, Etai Littwin, Shuangfei Zhai, Omid Saremi, Roni Paiss, and Joshua Susskind. The slingshot mechanism: An empirical study of adaptive optimizers and the grokking phenomenon. *arXiv preprint arXiv:2206.04817*, 2022.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Vikrant Varma, Rohin Shah, Zachary Kenton, János Kramár, and Ramana Kumar. Explaining grokking through circuit efficiency. *arXiv preprint arXiv:2309.02390*, 2023.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? In *International Conference on Learning Representations (ICLR)*, 2021.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of EMNLP*, 2018.

A Experiment Inventory

The released set contains 505 successful runs across 21 phases, about 1,070 GPU-hours on NVIDIA A100/H100-80GB. Every cell has at least two seeds; most have three, and the primary multi-hop curricula have five for the seed-variance analysis (Appendix J.9). A small number of runs failed on first launch because of environment or data-split issues and were re-run to completion; failed entries remain in the raw log and are excluded from every reported metric.

Of the 505 runs, 415 come from the curriculum sweep and its controls, and 90 are the exposure controls of the main text: 54 matched-exposure runs (six tasks $\times$ three orders $\times$ three seeds), 18 prefix-matched runs, and 18 baseline reruns. The phases are:

- A–C: primary curriculum sweeps on the three task groups.
- D–E: spectral features on all A/B/C checkpoints (about 5 CPU-hours) and the spectral forecaster.
- F.1–F.5: SAM, data-scale, model-depth, GrokFast and SAM-ρ sweeps.
- G.1–G.6: seed expansion, HotpotQA distractor and gold-paragraph, 355M replication, 12-layer from-scratch control.
- H.1–H.3: 1B replication (Pythia-1B), GSM8K transfer, MuSiQue replication.
- I.1, I.5, I.6: pre-transition spectral checkpoints, difficulty-metric ablation, MuSiQue 2-hop filtering.
- J.1, J.2, J.4: 50-step replication, chain-alignment ablation, spectral transfer to mod-113.

B Hyperparameters and Presentation Schemes

Group A (2-layer transformer). Causal self-attention and MLP blocks; $d_{\text{model}} = 128$, 4 heads, 2 layers, dropout 0. AdamW, learning rate 10^{-3}, weight decay 1.0, $\beta_1 = 0.9$, $\beta_2 = 0.98$. Batch 512, 100,000 steps, evaluation every 250 steps, gradient clipping at norm 1.0.

Group B (4-layer transformer). $d_{\text{model}} = 256$, 8 heads, 4 layers. AdamW, learning rate 3×10^{-4}, weight decay 0.1. Batch 256, 50,000 steps, evaluation every 250 steps.

Group C (pretrained GPT-2 125M). `openai-community/gpt2` (Radford et al., 2019), 12 layers, $d_{\text{model}} = 768$. Fine-tuned with AdamW, learning rate 5×10^{-5}, weight decay 0.01, effective batch 32 (micro-batch 8, gradient accumulation 4), 50 epochs (≈40,000 steps), evaluation every 250 steps. Difficulty is the number of supporting hops plus pretrained GPT-2 log-perplexity of the full input. The transition threshold for this group is validation accuracy 0.7, since terminal accuracy plateaus near 0.85–0.90 and chance is about 0.05.

Growing-pool curricula. At training fraction f, the sampler draws an index batch from a difficulty-quantile pool that grows linearly from the easiest (easy-first) or hardest (hard-first) 10% to the full distribution, $0.1 + 0.9f$. Difficulties are precomputed once per dataset. The anti-curriculum with split ratio s uses the hard-first law for $f < s$ and the easy-first law for $f \geq s$; the default is $s = 0.5$. *Alternating* draws batches alternately from the bottom and top 50%; *entropy-sorted* is easy-first ordered by input-token Shannon entropy.

Matched-exposure (global sort). The training set is tiled to the full step budget so every example appears the same number of times. The resulting multiset is ordered once: globally by difficulty, ascending or descending with ties broken at random, for the sorted arms, or by a single random permutation for the shuffled arm. Batches are drawn sequentially and without overlap, with no per-epoch replay. Per-example exposure over the whole run is identical across the three arms by construction; only the temporal order differs.

Prefix-matched (epoch sort). Training is divided into epochs, each one full pass over the training set, so every example is seen exactly once per epoch and exposure is equal at every epoch boundary. Within each epoch, examples are sorted easy→hard, hard→easy, or shuffled.

Transition time. t^* is the first evaluation step at which validation accuracy reaches the threshold (0.9 for Groups A and B, 0.7 for Group C). Thresholds were fixed before analysis. *Sharpness* is the largest single-interval increase in validation accuracy along a run; values ≥ 0.20 indicate a step-like transition and values ≤ 0.05 gradual improvement.

C Growing-Pool Curricula: Per-Task Values

Table 5 gives the growing-pool results on mod-97 addition, and Table 6 the GCD values used in the ordinal comparison of the main text. On GCD, two independent sets of baseline runs give the hard-first value (the split-ratio endpoint $s = 1$ and the GrokFast baseline); they agree to within 350 steps.

Table 5: Growing-pool curricula on mod-97 addition, mean t^* over three seeds.

Schedule	t^*
Uniform	1,083
Entropy-sorted, alternating	≈1,100
Anti-curriculum ($s = 0.5$)	24,500
Hard-first	55,333
Easy-first	64,000

Table 6: GCD: t^* under the three presentation schemes, in both directions. Growing-pool values come from the split-ratio sweep endpoints (Appendix F) and the GrokFast baseline runs (Appendix G).

Scheme	Easy-first	Hard-first
Epoch sort (prefix-matched)	583	750
Growing pool	8,000	15,583 (15,250 in GrokFast baseline)
Global sort (matched exposure)	12,250	27,250
Shuffled / uniform reference		500

D Spectral Dynamics

Across the 126 primary runs we compute the effective rank $\mathrm{er}(W) = \exp(-\sum_i \tilde{s}_i \log \tilde{s}_i)$, with $\tilde{s}$ the normalized singular values, for every weight matrix at every checkpoint. Runs under difficulty-restricted curricula show a sharp drop in effective rank at t^*, mirroring the jump in validation accuracy; runs under unrestricted sampling show a gradual decrease. This is consistent with the lazy-to-rich account of grokking (Kumar et al., 2024), in which generalization coincides with a collapse onto a compact feature representation. Restricted support delays the collapse.

E Data-Scale Ablation

On GCD, the curriculum-induced delay falls monotonically with training-set size N. At $N = 5$K it is 84× (uniform $t^* = 333$, hard-first 27,917, easy-first 31,500); at $N = 10$K it is 52×; at $N = 40$K it vanishes ($t^* \in \{250, 250, 500\}$ for the three curricula). A larger dataset makes every difficulty slice larger in absolute terms, so this shape is what a support account predicts.

F Split-Ratio Sweep

We parametrize the anti-curriculum by a split ratio $s \in [0, 1]$: hard-first for the first fraction s of training, then easy-first. On GCD (release group SR_B) the relation is non-monotonic with a minimum at $s = 0.25$: mean $t^* = 5{,}250$, against 8,000 at $s = 0$ and 15,583 at $s = 1$. A brief hard phase followed by easy consolidation is fastest. On 2WikiMultiHopQA with pretrained GPT-2 (release group SR_C) the relation is monotonic in s, and $s = 0$ is fastest by both transition time and terminal accuracy.

G Optimizer Baselines

SAM. We compare with SAM (Foret et al., 2021) ($\rho = 0.05$, non-adaptive), implemented as a two-step wrapper around AdamW, on 2 tasks × 3 curricula × 3 seeds. SAM leaves the uniform-curriculum trajectory unchanged but suppresses the sharp hard-first transition: the largest single-interval jump falls 9.3× (0.391 to 0.042), while t^* is unchanged or slightly later. SAM does not improve terminal accuracy. A sweep over $\rho \in \{0.01, 0.1, 0.2\}$ on mod-97 addition (9 runs) reaches terminal validation accuracy 1.000 but never restores the sharp transition; the largest jump stays ≤ 0.05 at every ρ.

GrokFast. GrokFast (Lee et al., 2024) amplifies low-frequency gradient components. We ran it with default settings ($\alpha = 0.98$, $\lambda = 2.0$) on mod-97 addition and GCD across the three primary curricula (18 runs). On GCD it speeds up easy-first from 8,000 to 4,833 steps (1.7×) and tightens the variance of the uniform curriculum, but hard-first stays at about 15,250 steps with or without it, so the ordering uniform < easy-first < hard-first is preserved. On mod-97 addition the effect is comparable: uniform and easy-first are within noise of the baseline, and hard-first is slightly faster on average. GrokFast shortens the smooth phase but does not move the ordering across curricula; the two interventions are composable.

H Spectral Forecaster: Details and Checks

Features and protocol. The forecaster uses the first standard checkpoint (step 1,000 for Groups A/B, step 500 for Group C) and chooses 2 of 10 singular-value statistics: effective rank and its slope, Frobenius norm and its slope, the ratio of the top two singular values, and the largest singular value. Because choosing the best of 45 pairs with 18 runs inflates in-sample fit, we cross-validate at two levels. Leave-one-run-out gives in-sample / held-out R^2 of 0.726 / 0.699 (A), 0.853 / 0.835 (B) and 0.922 / 0.890 (C). The main text reports the stricter fully nested leave-one-cell-out value, where the pair is chosen inside each fold and all seeds of a (task, curriculum) cell are held out together. Every run crossed its threshold, so no t^* is censored or imputed.

Pre-transition checkpoints. In Group C the fastest curricula cross the threshold before the step-500 checkpoint. We therefore re-ran the 18 Group C configurations for 300 steps, saving spectral state (excluding embeddings) at steps 25, 50, 100, 150, 200 and 250, and repeated leave-one-run-out selection over pairs from four features. The signal is present from step 25 ($R^2 = 0.72$, against 0.89 at step 500), and the same pair (top-two ratio, largest singular value) wins at both. R^2 falls at steps 200–250 because by then the uniform curricula have reached the threshold, so their $\log t^*$ collapses to a boundary value and the target becomes bimodal; the ranking of runs is unchanged. A matched training-loss baseline reaches only $R^2 = 0.14$ (leave-one-run-out) and -2.44 (nested).

Transfer to an unseen task. We pre-specified the pair (top-two ratio, largest singular value) and applied it without any feature selection to mod-113 addition, generated like mod-97 addition with a different modulus and vocabulary ($113^2 = 12{,}769$ pairs, 50/50 split, max-of-pair difficulty), with

6 curricula $\times$ 2 seeds. Refitting only the coefficients gives in-sample $R^2 = 0.756$, leave-one-out $R^2 = 0.593$ and Pearson $r = 0.870$ over $\log t^* \in [6.91, 11.40]$ (t^* from 1,000 to 89,000). The feature pair is not curve-fit to the original tasks, but we still have no single predictor that works across regimes.

Table 7: Spectral forecaster of $\log t^*$, R^2 under three evaluation protocols, 18 runs per regime. "Nested cell" chooses the feature pair inside each fold and holds out all seeds of a (task, curriculum) cell together; it is the number we rely on. The label-only baseline predicts from curriculum identity and cannot predict a held-out curriculum.

Regime	Modal feature pair	In-sample	Leave-one-run	Nested cell	Label only
A (modular)	Frob. norm, its slope	0.73	0.70	0.66	0.75
B (algorithmic)	Frob. slope, max SV	0.85	0.84	0.76	-0.77
C (pretrained)	top-2 ratio, max SV	0.92	0.89	0.81	-1.33

I Heuristic Intuition for Support Restriction

This appendix gives an informal argument for why a narrow support can delay the onset of a generalizing solution. It is an order-of-magnitude intuition, not a derivation, and no reported result depends on it.

Setting. Take inputs $x \in \{\pm 1\}^d$, a parity target $y(x) = \prod_{i \in S} x_i$ on a secret subset S of size k, and a network $\hat{y} = \sigma(\beta\, v^\top W x)$ with a smooth logistic nonlinearity. Training runs under gradient flow on logistic loss with weight decay λ, under a time-varying data distribution $\mathcal{D}_t$ with covariance $\Sigma_t = \mathbb{E}_{\mathcal{D}_t}[xx^\top]$ and target-aligned covariance $\Sigma_t^* = \mathbb{E}_{\mathcal{D}_t}[y(x)\, xx^\top]$ restricted to S. Define the alignment ratio $\kappa_t = \lambda_{\max}(\Sigma_t^{-1/2} \Sigma_t^* \Sigma_t^{-1/2})$.

Argument. Under $\dot{W} = -\nabla_W \mathcal{L}(\mathcal{D}_t) - \lambda W$, in the early small-margin regime the gradient linearizes to $\nabla_W \mathcal{L} \approx -v\, \mathbb{E}_{\mathcal{D}_t}[y(x) x^\top]$. Let $a(t)$ be the normalized alignment of $v^\top W(t)$ with the indicator of S. Informally, $\dot{a}(t) \le \kappa_t - \lambda a(t)$, and Grönwall's inequality gives $a(t) \le \int_0^t e^{-\lambda(t-s)} \kappa_s \, ds$: alignment with the target cannot grow faster than an exponential average of κ. The time at which a first reaches a small threshold ϵ is then, to leading order, at least ϵ divided by the time-averaged κ_t. A support restricted to one end of the difficulty range is poorly aligned with the target, keeps κ_t small, and so delays onset for as long as the restriction lasts. Making this a rigorous lower bound on t^* would need a two-sided control of the margin and of the nonlinearity; see Barak et al. (2022) for a rigorous treatment in a related setting.

J Multi-Hop QA Experiments with Pretrained Transformers

These experiments fine-tune pretrained language models on multi-hop question answering with growing-pool curricula. They are not exposure-controlled, and training loss is not near zero at the transition, so the jump is a curriculum-induced sharp validation transition rather than classical grokking. We report them as a map of where such a transition appears.

J.1 2WikiMultiHopQA: the sharp transition

On 2WikiMultiHopQA (Ho et al., 2020) with knowledge-graph evidence triples in the prompt (50,000 training examples, 80/20 split), uniform and easy-first schedules reach validation accuracy $\approx$0.85 at the first evaluation (step 250). Hard-first schedules instead hold validation accuracy near 0.33 (a floor reachable by reproducing training-answer statistics), then jump to $\approx$0.84 inside a 500-step window (steps 2,750–3,250; $t^* \in \{2{,}750, 3{,}000, 3{,}250\}$ across three seeds). The anti-curriculum at $s = 0.5$ transitions earlier, at 1,750 in all three seeds. Hard-first pays a terminal-accuracy cost of about 0.03–0.06 relative to easy-first.

Table 8: Hard-first curricula when fine-tuning pretrained models on multi-hop QA. Sharpness is the largest single-interval rise in validation accuracy (250-step evaluation).

Setting	Variation	Transition	Sharpness
2WikiMultiHopQA	GPT-2 125M	steps 2,750–3,250	0.27 (median)
	GPT-2 Medium 355M	steps 2,750–3,250	0.27–0.36
	Pythia-1B	same window	$\geq$0.28
	hop-count order only	step 2,750, all seeds	step-like
	perplexity-only or random order	none	–
	from scratch, 2–12 layers	none (acc. $\leq$0.06)	$\leq$0.02
HotpotQA (both settings)	GPT-2 125M	none	$\leq$0.06
MuSiQue (gold paragraphs)	GPT-2 125M	gradual recovery	–
GSM8K	GPT-2 125M	none (acc. $\approx$6%)	–

J.2 Pretraining versus depth

Trained from scratch with the same tokenizer, width ($d_{\text{model}} = 768$, 12 heads), data and schedule (learning rate 3×10^{-4}, Xavier initialization) at depths of 2, 4 and 6 layers, no curriculum produces the transition: terminal validation accuracy stays below 0.06 and the largest single-interval jump is ≤ 0.02 in all 9 configurations. A 12-layer from-scratch control, the same depth as the pretrained model, also fails. The pretrained 12-layer model produces the transition under hard-first with sharpness 0.266, four times the uniform curriculum's 0.066. Within our budget the missing ingredient is pretraining rather than depth. The from-scratch runs sit at the chance floor, so an optimization failure cannot be fully excluded.

J.3 Replication across model size

We replicated the three primary curricula with GPT-2 Medium (355M parameters, 24 layers, $d_{\text{model}} = 1024$) with identical data and hyperparameters, three seeds each (9 runs, 40,100 steps, about 70 GPU-hours). All three hard-first seeds show the floor phase and then a sharp transition at $t^* = 2{,}750$, 3,000 and 3,250, inside the same 500-step window as at 125M, with sharpness 0.27, 0.36 and 0.34 and terminal accuracy 0.91 ± 0.03. Uniform and easy-first reach ≥ 0.92 at the first evaluation. The hard-first transition also lands in the same window with Pythia-1B (Biderman et al., 2023).

Fine-grained replication. To test the window at finer resolution, we re-trained hard-first at all three sizes (GPT-2 125M, GPT-2 Medium 355M, Pythia-1B), three seeds each, evaluating every 50 steps to step 8,000, with the curriculum schedule matched to the original 50-epoch growth law (torch 2.5.1, transformers 4.57.6; 9 runs, about 45 GPU-hours). All 9 runs cross 0.7 inside a single 450-step window: 125M at 5,600, 5,700 and 5,850; 355M at 5,400, 5,500 and 5,600; 1B at 5,450, 5,550 and 5,750. Sharpness stays ≥ 0.28 at every size. Under this newer software stack the window sits about 2,500 steps later than in the original runs. Absolute timing is therefore implementation-dependent; what replicates is the narrow window shared across an $8\times$ range of parameters and two model families within a fixed stack.

J.4 Which difficulty signal matters

The Group C curriculum combines supporting-hop count with GPT-2 log-perplexity. We ran hard-first under three alternative signals, three seeds each. *Hops only* reproduces the floor-then-jump: $t^* = 2{,}750$ in all three seeds, a floor near 0.29 at steps 2,250–2,750, and terminal accuracy 0.819 ± 0.004. *Perplexity only* and *random difficulty* both cross 0.7 at the first evaluation and never show a floor (terminal 0.831 ± 0.038 and 0.847 ± 0.025). For reference, the combined metric gives mean $t^* = 3{,}083$ and terminal 0.840 ± 0.004. The pacing schedule is identical across these rows, so the transition depends on ordering by hop count, not on hard-first pacing as such. Because hop count is integer-valued, a new hop stratum enters the growing pool at a fixed step, which is consistent with the zero across-seed variance in t^*.

J.5 CHAIN-ALIGNMENT ABLATION

One hypothesis for 2WikiMultiHopQA's distinctive transition is lexical alignment of the bridge entity across its two supporting paragraphs. For each training example with a bridge entity (present in about 99.8% of rows), we replaced its mentions in the second supporting paragraph with a deterministic token-distinct paraphrase, leaving the first paragraph and the evidence triples unchanged, in 30%, 50% or 70% of training examples (validation left clean), and trained hard-first with three seeds per fraction. All 9 runs reproduce the floor near 0.29 through step 5,000 and then a sharp jump (mean ≥ 0.25 within one 250-step interval). Because terminal accuracy in these variants (0.714–0.765) sits close to the 0.7 threshold used elsewhere, we time the jump here at validation accuracy 0.5 ($t^* \approx 5{,}450$); this is the only place a different threshold is used, and it serves only to locate the jump. Terminal accuracy is 0.765 ± 0.017, 0.714 ± 0.009 and 0.741 ± 0.012 at 30%, 50% and 70% broken. Breaking bridge alignment lowers terminal accuracy but does not remove the transition, so paragraph-level bridge alignment is not the trigger. The evidence triples, which we left untouched, remain the most plausible candidate.

J.6 HOTPOTQA

With the identical pipeline on HotpotQA (Yang et al., 2018) in the distractor setting (three curricula, two seeds), all six runs plateau at exact match 0.29–0.31 (hard-first 0.300 ± 0.004, uniform 0.306 ± 0.000, easy-first 0.296 ± 0.004), with no sharp transition over about 59,400 steps, while training loss falls to 0.2–0.6. In the gold-paragraph setting (supporting paragraphs only; three curricula, three seeds), all 9 runs reach 0.67–0.70, about 2.3× higher, and terminal accuracy does not differ across curricula. None shows the 2WikiMultiHopQA transition: the largest single-interval jump is ≤ 0.06 in every run, against 0.27 for the median 2WikiMultiHopQA hard-first run. Removing distractors lifts HotpotQA off the floor but does not produce the transition.

J.7 MUSIQUE

MuSiQue (Trivedi et al., 2022) composes single-hop questions so that every answer requires chaining all supporting facts. We used the gold-paragraph split (19,938 train, 2,417 validation) with three curricula and three seeds. Terminal accuracy is 0.450 ± 0.021 (hard-first), 0.452 ± 0.032 (uniform) and 0.463 ± 0.021 (easy-first). Hard-first shows a floor phase: training loss falls below 0.01 by about step 2,500 while validation accuracy drops to 0.329 ± 0.006, close to the 2WikiMultiHopQA floor. Recovery is gradual, over about 20,000 steps, with no step-like jump. So the floor phase under hard-first appears on MuSiQue too, but the sharp transition does not.

Two-hop filtering. To test whether MuSiQue's mix of 2-, 3- and 4-hop questions prevents a sharp jump, we filtered to 2-hop questions only (14,376 train, 1,252 validation) and ran three curricula with three seeds (about 19,450 steps each). All three curricula converge to the same terminal accuracy (0.43–0.46), no run shows a floor, and no single-interval jump exceeds 0.08. Removing hop heterogeneity removes the curriculum effect rather than sharpening it, which again points to properties of 2WikiMultiHopQA beyond hop count, such as its explicit evidence triples, its uniform two-hop bridge structure, or its answer distribution.

J.8 GSM8K

On GSM8K (Cobbe et al., 2021), with difficulty given by the number of arithmetic steps in the reference solution (1–9, median 3), three curricula and three seeds on the 7,473-example training set give test exact match of $6.3 \pm 1.8\%$ (uniform), $5.8 \pm 1.6\%$ (hard-first) and $5.3 \pm 2.6\%$ (easy-first), statistically indistinguishable. The task is out of reach for GPT-2 125M at this data scale, so there is no transition to schedule. A fair test needs a larger model or a math task within reach of this one.

J.9 SEED EXPANSION

We expanded the primary 2WikiMultiHopQA curricula from three to five seeds. The terminal-accuracy ordering (uniform ≈ hard-first < easy-first) holds across all five. By Welch's t-test, hard-first is lower than easy-first ($p = 6.8 \times 10^{-4}$) and indistinguishable from uniform ($p = 0.48$).

Hard-first solutions are highly reproducible in terminal accuracy (standard deviation 0.007), while easy-first varies slightly more (0.016) but is consistently higher.