DEMYSTIFYING AND ACCELERATING EMERGENCE TO NEW TASKS IN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) has shown strong gains in complex reasoning for large language models, but a central question remains whether current post-training techniques truly incentivize models to generalize to new tasks, or whether they mainly reinforce patterns already present in the training data. In this work, we take a mechanistic perspective on this question by investigating a phenomenon called delayed emergence, where a policy first overfits observed rollouts through memorization and much later transitions to out-of-distribution generalization. By analyzing the circuits during training, we show that this capability transition is characterized by a non-monotonic trajectory in the proposed circuit rank and orthogonality index, reflecting a shift from memorization circuits to generalizing circuits. We further identify a critical bottleneck in the RLVR gradient, where low reward variance on training prompts can weaken the signal for transferable features and delay generalization. Based on this insight, we propose Directed Feature Emergence (DFE), a spectral intervention that provably encourages learning on the weak directions in the circuit while preserving the useful representation. Experiments on DELTA-Code, including Manufactoria and BouncingSim, show that DFE consistently accelerates delayed generalization under PPO and GRPO across Qwen3 models, especially in extreme out-of-distribution settings where the base model obtains nearly zero reward. Our results suggest that circuit-level dynamics provide both a diagnostic view of when RLVR moves beyond memorization and a practical target for accelerating task emergence.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has shown great promise in improving the reasoning capabilities of large language models (Shao et al., 2024; DeepSeek-AI, 2025). By optimizing against a binary correctness signal, RLVR has produced striking gains in mathematical and scientific reasoning, often outperforming supervised fine-tuning (SFT) under the same data distribution. Beyond data-centric improvements, RLVR is increasingly seen as a path toward models that can generalize, self-evolve, and autonomously discover new task capabilities through exploration and exploitation. Following GRPO (Shao et al., 2024), many variants of reinforcement learning have been proposed to push the limits of algorithmic improvement. Yet a central question remains: *does reinforcement learning truly induce models to generalize to new tasks, or does it simply reinforce patterns already present in the training data?*

This question is more important than it may seem. If RLVR only reinforces patterns already present in the training distribution, its promise as a path toward open-ended capability discovery is fundamentally limited, and scaling it further will not lead to qualitatively new behaviors. Two observations make this concern concrete. First, RL and SFT produce qualitatively different generalization behaviors even at the same training accuracy, with RL trained models generalizing to new task variants while supervised models fail to do so (Chu et al., 2025). Second, the benefit of RLVR is highly inconsistent across tasks and training configurations, and the same algorithm can succeed or fail depending on the structure of the problem and the policy objective (Yue et al., 2025; Chen et al., 2025). This suggests that, in some cases, RLVR does not uniformly improve all capabilities but rather triggers a qualitative change in some cases and not others. Understanding what determines this outcome is both a fundamental scientific question and a practically important one.

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.

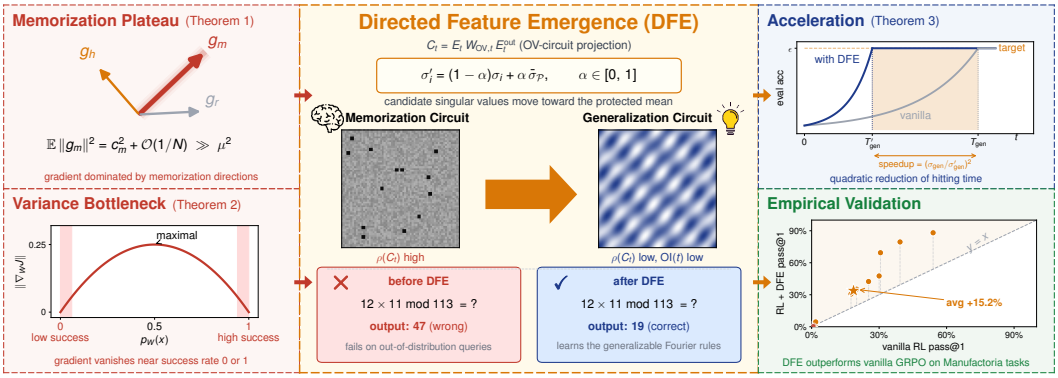


Figure 1: Conceptual illustration of delayed emergence in RLVR (Theorem 1 and 2) and the proposed DFE intervention for accelerating the transition from memorization to generalization (Theorem 3).

In the supervised learning setting, there is a widely-studied phenomenon deeply related to this problem, namely "grokking" or delayed generalization (Power et al., 2022; Nanda et al., 2023; Gromov, 2023). These works study how supervised models transition from memorization to generalization under continued training. Prior work has shown that this transition is driven by the weight decay favoring a more parameter-efficient generalizing circuit over the memorizing regime (Nanda et al., 2023; Varma et al., 2023; Liu et al., 2022) in synthetic mathematical settings such as modular arithmetic and permutation. Omnigrok (Liu et al., 2023) takes a step further to the grokking phenomenon on general image, text and graph data. Most recently, a benchmark paper (Sun et al., 2026) highlights the urgency of demystifying and resolving the grokking phenomenon in the reinforcement learning setting, but underscoring what mechanisms inducing task emergence under RLVR remain poorly understood.

In this work, we take a mechanistic perspective on delayed emergence in RLVR, as shown in Figure 1. We show that early RL training can enter a memorization plateau, where the policy improves training reward by fitting observed rollouts but fails to learn the underlying rules. We identify a variance bottleneck in the RLVR objective. When rollouts for a training prompt almost always succeed or fail, limited reward contrast can suppress the gradient for learning generalizing features. We then characterize emergence as a circuit-level transition in the OV-circuit matrix (Elhage et al., 2021), measured through changes in circuit rank and the orthogonality of task-token representations. Based on this mechanism, we propose Directed Feature Emergence (DFE), a spectral intervention that preserves dominant output-projection components while encouraging weak candidate directions to accelerate this transition. Our analysis connects weak training signals to delayed transfer and motivates an intervention on the underlying circuit.

Our contributions are as follows:

- We provide the first mechanistic characterization of the delayed emergence to new tasks in RLVR, connecting the variability of RL generalization outcomes to a circuit-level phase transition characterized by changes in circuit rank, spectral concentration, and representation orthogonality, as shown in Figure 2.

- Based on this mechanistic characterization, we propose a novel approach Directed Feature Emergence (DFE) that can accelerate the phase transition at the early stage of training, significantly improving generalization on fully out-of-distribution tasks where the base model achieves nearly zero rewards.

- Extensive experiments on various baseline models and tasks from the modular arithmetic task and DELTA-CODE benchmark (Sun et al., 2026) demonstrate the effectiveness of the proposed approach and offer a new perspective on the importance of task emergence in reinforcement learning post-training.

2 Related Work

Phase transitions and delayed emergence. Early work on grokking, or delayed generalization (Power et al., 2022), showed that transformers trained on modular arithmetic first memorize the training data and generalize much later, establishing that the two regimes correspond to mechanistically distinct solutions. Subsequent work identified the transition as the emergence of a sparse Fourier circuit in the modular addition task, driven by weight decay favoring a more parameter-efficient generalizing solution over the memorizing one (Nanda et al., 2023; Liu et al., 2022; Gromov, 2023). Similar transitions have been observed in parity learning (Barak et al., 2022) and in multi-task settings where generalizing solutions lie on low-dimensional manifolds (Xu, 2026). However, these works mainly focus on supervised learning and treat weight decay as the to-go solution. These works do not address whether or how such transitions can be triggered under reinforcement learning with verifiable rewards. We extend this line of work to the RLVR setting, identify circuit dynamics as the guiding mechanism for emergence, and show how this insight leads to a practical method for accelerating the transition.

Reinforcement learning with verifiable rewards. RLVR improves mathematical reasoning without supervised demonstrations (DeepSeek-AI, 2025), using GRPO (Shao et al., 2024), PPO (Schulman et al., 2017), and related methods (Liu et al., 2025b). DAPO addresses ineffective updates and entropy collapse through dynamic sampling and asymmetric clipping (Yu et al., 2025). Studies examine whether RLVR teaches new skills (Chu et al., 2025; Yue et al., 2025; Wang et al., 2026) and how KL regularization and entropy affect generalization (Shenfeld et al., 2025; Chen et al., 2025; Wang et al., 2025). ProRL shows that prolonged RL can expand reasoning capabilities (Liu et al., 2025a), while Fan et al. (2026) analyze how sampling bias and semantic coupling suppress alternative solutions. Complementing these studies, we examine circuit dynamics underlying delayed transfer and introduce a spectral intervention to accelerate its emergence. Our local feature-learning analysis draws on grokking theory (Tian, 2026).

Mechanistic interpretability. Mechanistic interpretability aims to understand the internal algorithms of neural networks at the level of circuits and features (Sharkey et al., 2025; Elhage et al., 2021). Applied to grokking, this approach identified the specific Fourier circuits responsible for modular arithmetic and showed that their emergence marks the onset of generalization (Nanda et al., 2023). Work on intrinsic dimensionality (Li et al., 2018) has further shown that generalizing solutions are inherently low-dimensional, a property also observed in modular exponentiation (Africa et al., 2025). However, existing work focuses on characterizing the generalizing circuit after the transition occurs and does not study the dynamics that govern whether and when the transition is triggered under *reinforcement learning*. Our work bridges this gap by analyzing circuit-level dynamics throughout RLVR training and using the principled understanding to motivate a method that accelerates emergence.

3 Circuit Dynamics of Emergence

Delayed emergence occurs when training performance and out-of-distribution performance rely on distinct intrinsic computations (memorization versus generalization). Following previous works (Power et al., 2022; Gromov, 2023), we start from modular arithmetic as a controlled setting to observe the circuit-level changes. The model can memorize observed input-output pairs, or it can form the generalizable modular arithmetic knowledge (e.g., Fermat's Little Theorem), which can be applied to infinitely more unseen samples. These two distinct solutions correspond to separate directions in the Output-Value circuit (Elhage et al., 2021). Below, we formally define the circuit dynamics of delayed emergence.

Memorization and Generalization Circuits Following prior work on delayed generalization (Power et al., 2022; Nanda et al., 2023; Gromov, 2023), we study modular arithmetic over the cyclic group $\mathbb{Z}_p$. For a large prime number p and operation $\circ \in \{+, \times\}$, each example has the form $a \circ b \equiv c \pmod{p}$. The vocabulary therefore contains p task tokens with the special tokens for operations. We study this controlled setting in a two-layer transformer architecture using supervised fine-tuning (SFT), Proximal Policy Optimization (PPO) (Schulman et al., 2017) and Group Relative Policy Optimization (GRPO) (Shao et al., 2024). More implementation details are in the Appendix. In

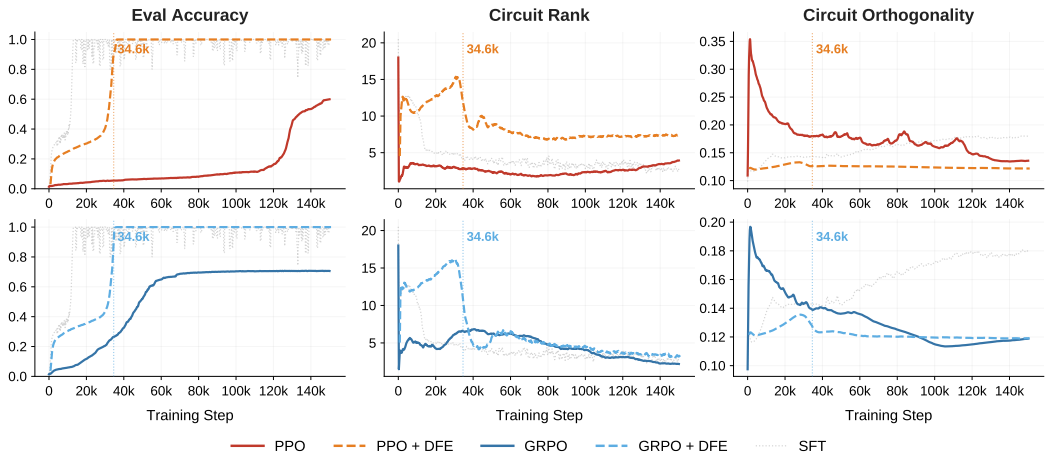


Figure 2: Circuit dynamics on modular multiplication ($p = 113$) under SFT, PPO, and GRPO, including RL runs with DFE. The three columns show the evaluation accuracy, the circuit rank of the OV circuit, and the orthogonality index of the task-token representations, all over training. The marked transition under DFE occurs at around step 34.6k.

Figure 2, we observe that the model first fits the training distribution by memorizing instance-specific associations. After many steps, it suddenly generalizes to the full distribution by representing the true algebraic computation that maps any pair to the correct answer, which we refer to *phase transition*. We define the first solution as the model activating the *memorization circuit* and the second as the model activating the *generalization circuit*. This distinction can be shown in evaluation accuracy since the out-of-distribution evaluation distribution differs from the training distribution at the instance level while sharing the same underlying computational rules.

Geometry of OV Circuits We measure the representation change through the scope of the Output-Value (OV) circuit (Elhage et al., 2021). In the standard transformer architecture (Vaswani et al., 2017), W_V denotes the value projection matrix in the attention head, which maps the input token representations into value vectors. The matrix W_O denotes the output projection matrix, which maps the value vectors back to the model residual stream. Following the standard OV circuit view of transformers (Elhage et al., 2021), we define the composed value-output matrix as $W_{\mathrm{OV}} = W_V \cdot W_O$. At training step t, the task-projected OV circuit is defined as

$$C_t = E_t \cdot W_{\mathrm{OV},t} \cdot E_t^{\mathrm{out}} \in \mathbb{R}^{n \times n}, \tag{1}$$

where n is the number of task tokens, $E_t \in \mathbb{R}^{n \times d}$ contains the input embeddings, and $E_t^{\mathrm{out}} \in \mathbb{R}^{d \times n}$ is the corresponding output embedding. For simplicity, when there is no ambiguity, we omit the dependence on t. This matrix C_t reveals how attended task-token information converts into output-logit directions, which exposes the internal mechanism of the learned representation. Below, we introduce two practical measurements. Their interpretation is discussed in Appendix D.1.

Definition 1. For a matrix W with singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$, let $q_i = \sigma_i^2 / \sum_j \sigma_j^2 > 0$. We define the **circuit rank** $\rho(W)$ and the **orthogonality index** $\mathrm{OI}(t)$ as follows.

$$\rho(W) = \exp\Big(-\sum_i q_i \log q_i\Big), \quad \mathrm{OI}(t) = \frac{\|\hat{E}_t \hat{E}_t^\top - I_n\|_F}{\sqrt{n(n-1)}}, \tag{2}$$

where $\hat{E}_t$ is the row-normalized task-token embedding and $\|\cdot\|_F$ denotes the Frobenius norm.

We focus on $\rho(C_t)$ throughout the analysis. Lower OI indicates that task-token representations are more orthogonally separated. As illustrated in Figure 2, the transition between memorization and generalization is characterized by a distinct, non-monotonic trajectory in the circuit rank $\rho(C_t)$. It first contracts during memorization, expands as generalization directions gain energy, and contracts again as the final representation compacts.

4 Delayed Emergence in Reinforcement Learning

To explain how reinforcement learning induces capability emergence and why this process can be delayed, we decompose the policy gradient over the OV circuit into orthogonal feature subspaces and analyze its finite-batch concentration (Jacot et al., 2018; Tian, 2026). Within this framework, Theorem 1 characterizes the gradient regime that produces the memorization plateau, Theorem 2 identifies the variance bottleneck that scales the policy gradient, and Theorem 3 gives conditions under which spectral amplification accelerates emergence, motivating our proposed method, Directed Feature Emergence (DFE).

4.1 Memorization Plateau in Early RL training

We analyze the early stage of RL fine-tuning through the local geometry of the OV circuit. Let W denote the OV-circuit matrix. For a finite RL batch $\{(x_i, y_i, A_i)\}_{i=1}^N$, where A_i is the advantage of completion y_i on prompt x_i and is bounded ($|A_i| \le 1$), the empirical KL-regularized objective (Schulman et al., 2017; Shao et al., 2024) has an initial gradient

$$g = \frac{1}{N}\sum_{i=1}^{N} A_i\,\phi_i, \quad \phi_i = \nabla_W \log \pi_{W_0}(y_i \mid x_i).$$

The initial RL update is therefore an advantage-weighted combination of rollout-induced features. Since A_i is bounded and does not affect asymptotic rates, we absorb it into the features below and write m_i, h_i, r_i for the corresponding advantage-weighted components.

We write $\mathcal{M}$, $\mathcal{G}$, and $\mathcal{R}$ for memorization, generalization, and irrelevant feature subspaces, with corresponding gradient components g_m, g_h, and g_r. The decomposition and statistical assumptions are stated in Appendix C.1.

At initialization, each ϕ_i may carry mass on all three subspaces. With only finitely many rollouts, the projection onto the true rule direction is weak and noisy, since several candidate rules are consistent with the batch. As N grows, candidate-rule noise averages out and the true-rule signal becomes accurately estimated.

Theorem 1 (Memorization Plateau in Early RL)**.** *Under Assumptions 1 and 2, let $c_m = \|\mathbb{E}[m_i]\| > 0$ and $\mu = \mathbb{E}[\langle h_i, u\rangle]$ for the unit rule direction $u \in \mathcal{G}$. Then*

$$\mathbb{E}\|g_m\|^2 = c_m^2 + \mathcal{O}(1/N), \qquad \mathbb{E}\|g_h\|^2 = \mu^2 + \mathcal{O}(1/N), \qquad \mathbb{E}\|g_r\|^2 = \mathcal{O}(1/N).$$

When $|\mu| \ll c_m$ and the sampling terms are small, the expected gradient energy is concentrated in memorization directions. This gives a local account of the imbalance associated with the *memorization plateau*. Increasing N reduces estimation noise but does not change the population alignment μ. The result characterizes the initial gradient rather than the duration of the plateau. A formal statement and proof are deferred to Appendix C.

4.2 Policy Gradient Dynamics and the Variance Bottleneck

Theorem 1 describes an initial regime in which memorization directions dominate when $|\mu| \ll c_m$. We show that the RLVR objective itself provides a mechanism to amplify this signal, but only through a strict variance bottleneck. Let the policy be a softmax over completions, $\pi(y \mid x) = {\exp(f_W(x,y))}/{\sum_{y'}\exp(f_W(x,y'))}$, with W as the OV-circuit matrix. Let $\mathcal{D}$ be the training prompt distribution. The RLVR objective maximizes the expected verifier reward

$$J(W) \;=\; \mathbb{E}_{x\sim\mathcal{D}}\,\mathbb{E}_{y\sim\pi(\cdot|x)}[v(x,y)], \qquad v(x,y) \in \{0,1\}.$$

For each prompt x, let $p_W(x) := \mathbb{P}_{y\sim\pi(\cdot|x)}(v(x,y)=1)$ denote the success probability.

Theorem 2 (Variance-Gated Policy Gradient)**.** *The expected RLVR gradient with respect to the OV-circuit matrix W admits the closed form*

$$\nabla_W J(W) \;=\; \mathbb{E}_{x\sim\mathcal{D}}\Big[\,p_W(x)\big(1-p_W(x)\big)\,\Delta\phi(x)\Big], \tag{3}$$

where the success/failure feature contrast $\Delta\phi(x)$ *is*

$$\mathbb{E}_{y\sim\pi(\cdot|x)}\big[\nabla_W \log\pi(y\mid x)\,\big|\,v(x,y)=1\big] \;-\; \mathbb{E}_{y\sim\pi(\cdot|x)}\big[\nabla_W \log\pi(y\mid x)\,\big|\,v(x,y)=0\big].$$

The proof is in Appendix C. Theorem 2 separates the feature contrast $\Delta\phi(x)$ from the binary reward variance $p_W(x)(1 - p_W(x))$. For bounded feature contrast, this *variance bottleneck* suppresses updates when a training prompt almost always succeeds or almost always fails. Learning transferable features further requires the contrast to align with $\mathcal{G}$. Thus, high reward on memorized training prompts can coexist with slow growth of generalizing features. Low held-out success measures the resulting transfer gap and does not enter the training gradient. Appendix C.3 relates this analysis to the experimental objectives.

5 Directed Feature Emergence

As established in Section 4.2, limited reward contrast on training prompts can slow the growth of transferable features, even when training reward is high. We introduce Directed Feature Emergence (DFE), a spectral intervention that strengthens weak candidate directions before RL resumes. The method is summarized in Algorithm 1 in Appendix E.

Spectral Intervention. DFE edits the output projection W_O of the targeted attention layer, which changes the composed OV circuit $W_{\mathrm{OV}} = W_V W_O$. We hypothesize that while the generalization subspace $\mathcal{G}$ is suppressed by the variance bottleneck, useful directions overlap the low-energy components of W_O. At the intervention step (Appendix F), we compute the Singular Value Decomposition (SVD) of the output projection $W_O = \tilde{U}\Sigma\tilde{V}^\top$, with singular values $\sigma_1 \geq \cdots \geq \sigma_r$ as in Definition 1. We then partition the spectrum into *protected directions* $\mathcal{P} = \{1, \ldots, k\}$ and *candidate directions* $\mathcal{B} = \{k{+}1, \ldots, r\}$. The index k is chosen such that the protected dimensions explain a dominant fraction of the total spectral energy. The protected set $\mathcal{P}$ retains the dominant components of the current output projection. We amplify the candidate set $\mathcal{B}$ to strengthen weak directions that may support generalization.

Acceleration of Emergence. To accelerate emergence, the intervention increases energy in the candidate subspace $\mathcal{B}$. We first compute the mean of the protected singular values $\bar{\sigma}_{\mathcal{P}}$. For each candidate direction $i \in \mathcal{B}$, DFE modifies its singular value as

$$\sigma_i' = (1 - \alpha)\sigma_i + \alpha\bar{\sigma}_{\mathcal{P}}, \qquad \alpha \in [0, 1], \tag{4}$$

where α is the intervention strength. We reconstruct W_O with the modified spectrum Σ' and load it back into the policy π_t to resume training. This preserves the protected components exactly and bounds the change in spectral norm by $\alpha\bar{\sigma}_{\mathcal{P}}$.

The edit also changes the composed circuit $W_{\mathrm{OV}} = W_V W_O$. With W_V fixed, every ordered singular value of W_{OV} is nondecreasing, and energy increases strictly along directions that overlap an amplified candidate component. The derivation is given in Appendix C. The following theorem quantifies how this amplification accelerates an aligned generalization coordinate when it exceeds the residual error in the local dynamics.

Theorem 3 (Acceleration of DFE)**.** *Under Assumption 3 in Appendix C.1, let T and T' be the times for the original and edited generalization coordinates to reach the same target, with corresponding singular values $s < s'$, effective learning rate η, and residual bound δ. If $\eta((s')^2 - s^2) > 2\delta$, then*

$$\frac{T'}{T} \leq \frac{\eta s^2 + \delta}{\eta (s')^2 - \delta} < 1. \tag{5}$$

For $\delta = 0$, the ratio is exactly $s^2/(s')^2$.

The proof is provided in Appendix C. Theorem 3 extends the inverse-square hitting-time relation to nonzero residual dynamics. It gives an explicit amplification margin that guarantees faster emergence under the stated local dynamics. The appendix further relates this margin to the intervention strength α and the overlap with the edited directions.

6 Experiments

6.1 Implementation Details

Benchmarks. DELTA-Code (Sun et al., 2026) is a benchmark of out-of-distribution code-synthesis tasks designed to test whether RLVR can produce models that generalize beyond their training

Table 1: Pass@1 (%) on Manufactoria (Sun et al., 2026) across four difficulty tiers. Results use three seeds. Best per model size in **bold**.

	Basic			Easy				Medium		Hard			
Model	App.	Exa.	Sta.	Has	Reg.	End.	Com.	Prep.	BitOp	Symm	MnMx	Add	**Avg**
Oracle Methods													
GPT-5	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$92.0_{\pm 1.08}$	$92.0_{\pm 0.82}$	$95.0_{\pm 1.10}$	$80.0_{\pm 1.31}$	$20.0_{\pm 1.69}$	$30.0_{\pm 1.66}$	$25.0_{\pm 1.47}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$61.2_{\pm 0.51}$
o4-mini	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$55.0_{\pm 2.87}$	$55.0_{\pm 2.47}$	$85.0_{\pm 1.76}$	$30.0_{\pm 1.47}$	$5.0_{\pm 0.73}$	$2.0_{\pm 0.47}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$44.3_{\pm 0.72}$
Claude-Sonnet-4	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$65.0_{\pm 1.56}$	$70.0_{\pm 1.65}$	$75.0_{\pm 1.37}$	$30.0_{\pm 1.76}$	$5.0_{\pm 0.87}$	$5.0_{\pm 0.92}$	$3.0_{\pm 0.96}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$46.1_{\pm 0.50}$
Qwen3-235B-A22B	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$100.0_{\pm 0.00}$	$80.0_{\pm 1.86}$	$80.0_{\pm 1.33}$	$75.0_{\pm 1.66}$	$45.0_{\pm 1.64}$	$5.0_{\pm 0.68}$	$12.0_{\pm 1.68}$	$2.0_{\pm 0.63}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$49.9_{\pm 0.20}$
Qwen3-1.7B													
Base	$4.6_{\pm 1.05}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.4_{\pm 0.09}$
+ PPO	$23.5_{\pm 1.79}$	$16.8_{\pm 1.45}$	$10.6_{\pm 1.73}$	$7.4_{\pm 1.14}$	$5.0_{\pm 0.69}$	$9.0_{\pm 1.06}$	$5.7_{\pm 1.00}$	$0.5_{\pm 0.25}$	$0.0_{\pm 0.00}$	$0.1_{\pm 0.05}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$6.6_{\pm 0.37}$
+ GRPO	$28.4_{\pm 1.75}$	$19.5_{\pm 1.51}$	$13.8_{\pm 1.51}$	$10.5_{\pm 1.52}$	$7.5_{\pm 0.98}$	$12.6_{\pm 0.87}$	$8.5_{\pm 1.30}$	$0.6_{\pm 0.30}$	$0.4_{\pm 0.20}$	$0.2_{\pm 0.10}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$8.5_{\pm 0.33}$
+ PPO + **DFE**	$36.7_{\pm 1.35}$	$24.6_{\pm 1.69}$	$19.0_{\pm 1.35}$	$14.5_{\pm 1.00}$	$10.5_{\pm 0.96}$	$17.0_{\pm 1.54}$	$12.0_{\pm 1.18}$	$1.4_{\pm 0.67}$	$0.7_{\pm 0.25}$	$0.3_{\pm 0.15}$	$1.3_{\pm 0.45}$	$0.0_{\pm 0.00}$	$11.5_{\pm 0.31}$
+ GRPO + **DFE**	$\mathbf{45.3}_{\pm 1.73}$	$\mathbf{33.8}_{\pm 2.46}$	$\mathbf{26.4}_{\pm 1.59}$	$\mathbf{18.7}_{\pm 2.18}$	$\mathbf{13.6}_{\pm 1.61}$	$\mathbf{21.5}_{\pm 1.37}$	$\mathbf{15.5}_{\pm 1.18}$	$\mathbf{2.2}_{\pm 0.55}$	$\mathbf{1.0}_{\pm 0.36}$	$\mathbf{0.4}_{\pm 0.20}$	$\mathbf{2.6}_{\pm 0.62}$	$\mathbf{0.4}_{\pm 0.20}$	$\mathbf{15.1}_{\pm 0.48}$
Qwen3-4B													
Base	$28.5_{\pm 1.61}$	$13.4_{\pm 1.08}$	$8.8_{\pm 1.12}$	$0.6_{\pm 0.23}$	$0.4_{\pm 0.20}$	$0.7_{\pm 0.35}$	$0.3_{\pm 0.15}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.1_{\pm 0.05}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$4.4_{\pm 0.31}$
+ PPO	$46.6_{\pm 2.31}$	$33.5_{\pm 2.33}$	$27.2_{\pm 1.74}$	$18.5_{\pm 1.52}$	$12.6_{\pm 2.02}$	$22.4_{\pm 1.65}$	$14.5_{\pm 2.25}$	$1.1_{\pm 0.40}$	$1.4_{\pm 0.42}$	$0.5_{\pm 0.25}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$14.9_{\pm 0.69}$
+ GRPO	$54.0_{\pm 1.89}$	$39.3_{\pm 1.30}$	$30.6_{\pm 1.97}$	$25.3_{\pm 1.50}$	$17.5_{\pm 1.06}$	$30.0_{\pm 1.84}$	$20.0_{\pm 2.27}$	$1.6_{\pm 0.50}$	$1.7_{\pm 0.51}$	$\mathbf{1.0}_{\pm 0.46}$	$1.8_{\pm 0.42}$	$\mathbf{0.7}_{\pm 0.35}$	$18.6_{\pm 0.76}$
+ PPO + **DFE**	$73.6_{\pm 2.05}$	$62.4_{\pm 1.93}$	$51.0_{\pm 2.05}$	$33.6_{\pm 1.52}$	$24.0_{\pm 3.34}$	$38.5_{\pm 2.02}$	$27.5_{\pm 1.97}$	$2.9_{\pm 0.63}$	$2.0_{\pm 0.55}$	$0.8_{\pm 0.26}$	$3.0_{\pm 0.76}$	$0.4_{\pm 0.20}$	$26.6_{\pm 0.86}$
+ GRPO + **DFE**	$\mathbf{88.1}_{\pm 1.33}$	$\mathbf{79.5}_{\pm 1.58}$	$\mathbf{69.4}_{\pm 1.77}$	$\mathbf{42.4}_{\pm 1.69}$	$\mathbf{30.7}_{\pm 2.11}$	$\mathbf{47.6}_{\pm 1.41}$	$\mathbf{35.2}_{\pm 1.49}$	$\mathbf{4.3}_{\pm 0.89}$	$\mathbf{2.6}_{\pm 0.56}$	$0.9_{\pm 0.33}$	$\mathbf{4.5}_{\pm 0.90}$	$0.5_{\pm 0.25}$	$\mathbf{33.8}_{\pm 0.70}$
Qwen3-8B													
Base	$42.7_{\pm 2.03}$	$23.2_{\pm 1.84}$	$15.3_{\pm 1.70}$	$1.4_{\pm 0.55}$	$0.8_{\pm 0.40}$	$1.4_{\pm 0.43}$	$0.5_{\pm 0.22}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.2_{\pm 0.10}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$7.1_{\pm 0.40}$
+ PPO	$61.5_{\pm 1.87}$	$46.0_{\pm 1.64}$	$38.6_{\pm 2.64}$	$25.2_{\pm 1.67}$	$17.1_{\pm 1.74}$	$33.3_{\pm 2.12}$	$20.2_{\pm 1.83}$	$1.6_{\pm 0.33}$	$2.4_{\pm 0.53}$	$0.8_{\pm 0.27}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$20.6_{\pm 0.51}$
+ GRPO	$68.7_{\pm 2.47}$	$52.7_{\pm 2.48}$	$41.9_{\pm 1.99}$	$36.2_{\pm 1.34}$	$24.7_{\pm 1.85}$	$42.3_{\pm 1.36}$	$27.4_{\pm 2.30}$	$2.3_{\pm 0.67}$	$3.0_{\pm 0.56}$	$\mathbf{1.6}_{\pm 0.54}$	$3.3_{\pm 0.84}$	$\mathbf{1.4}_{\pm 0.45}$	$25.5_{\pm 0.78}$
+ PPO + **DFE**	$85.6_{\pm 1.59}$	$78.3_{\pm 1.18}$	$67.5_{\pm 1.91}$	$46.2_{\pm 2.30}$	$34.0_{\pm 1.55}$	$52.7_{\pm 2.01}$	$37.9_{\pm 2.21}$	$4.1_{\pm 1.01}$	$3.1_{\pm 0.67}$	$1.4_{\pm 0.41}$	$4.5_{\pm 1.12}$	$0.6_{\pm 0.30}$	$34.7_{\pm 0.65}$
+ GRPO + **DFE**	$\mathbf{95.8}_{\pm 0.75}$	$\mathbf{91.9}_{\pm 0.99}$	$\mathbf{84.2}_{\pm 1.37}$	$\mathbf{57.1}_{\pm 2.92}$	$\mathbf{40.5}_{\pm 1.62}$	$\mathbf{62.8}_{\pm 2.15}$	$\mathbf{48.3}_{\pm 2.05}$	$\mathbf{6.1}_{\pm 0.60}$	$\mathbf{4.0}_{\pm 0.76}$	$1.2_{\pm 0.40}$	$\mathbf{5.9}_{\pm 0.79}$	$0.6_{\pm 0.28}$	$\mathbf{41.5}_{\pm 0.63}$

distribution. It consists of two tasks. **Manufactoria** evaluates the model's capability to write programs in a tape-manipulation domain-specific language (DSL) across 12 sub-tasks in four difficulty tiers. The Basic tier covers append (`App.`), exact-match (`Exa.`), and start-match (`Sta.`). The Easy tier covers substring containment (`Has`), regular-expression matching (`Reg.`), suffix matching (`End.`), and threshold comparison (`Com.`). The Medium tier covers sequence prepending (`Prep.`) and bitwise operations (`BitOp`). The Hard tier covers symmetric counting (`Symm`), min-or-max with a constant (`MnMx`), and constant addition (`Add`). **BouncingSim** evaluates the model to predict bouncing-ball trajectories under various physics rules, organized into six single-skill families (`RotObj`, `RotBox`, `MovBox`, `Gravity`, `MultiBox`, `MultiObj`), each instantiated at five difficulty levels from Basic to Extreme. We report Manufactoria results per sub-task and BouncingSim results per family. More details are provided in Appendix G and F.

Training setup. Each RL row trains Qwen3-1.7B, Qwen3-4B and Qwen3-8B with PPO (Schulman et al., 2017) or GRPO (Shao et al., 2024) as the policy-gradient estimator, with and without the DFE edit. The comparison X **+ DFE** vs. X for $X \in \{\text{PPO}, \text{GRPO}\}$ is matched on every other hyperparameter so that any difference is attributable to the spectral edit alone. Full implementation details are deferred to Appendix F.

6.2 Main Results

The main comparison is X **+ DFE** versus X within each model block. We observe three consistent patterns across Tables 1 and 2. First, on the Basic and Easy Manufactoria sub-tasks where vanilla RL already improves pass@1 above the base model, DFE yields a substantial further improvement. The 4B GRPO+DFE row reaches 88.1% on `App.`, 79.5% on `Exa.`, and 69.4% on `Sta.`, against 54.0, 39.3, and 30.6 for vanilla GRPO, and the same pattern holds across the four Easy sub-tasks. Second, on the Medium and Hard tiers where vanilla PPO and GRPO collect almost no reward within the compute budget, the relative gains are largest. The 4B GRPO+DFE row attains 4.5% on `MnMx` and 4.3% on `Prep.`, more than twice the vanilla GRPO numbers; the only exceptions are two of the smallest-signal Hard sub-tasks (`Symm` and `Add`) where vanilla GRPO holds a marginal lead. Third, BouncingSim shows the same pattern at every family, with the strongest absolute improvements on `Gravity` (22.5 vs. 13.5) and `MultiObj` (22.0 vs. 14.0), the two families on which the base policy already finds occasional perfect rollouts. The improvement is consistent between the PPO and GRPO methods across both benchmarks, indicating that DFE improves final performance on these out-of-distribution tasks. Training speed is examined in Section 6.4.

Table 2: Pass@1 (%) on BouncingSim (Sun et al., 2026), averaged over five difficulty tiers per physics family under 3 random seeds. Best per column within each model size group in **bold**.

Model	RotObj	RotBox	MovBox	Gravity	MultiBox	MultiObj	Avg
Oracle Methods							
GPT-5	$15.0_{\pm 1.53}$	$34.8_{\pm 2.77}$	$39.6_{\pm 1.87}$	$78.8_{\pm 2.39}$	$9.8_{\pm 1.09}$	$29.6_{\pm 1.86}$	$34.6_{\pm 0.79}$
o4-mini	$6.2_{\pm 1.40}$	$7.6_{\pm 0.96}$	$7.0_{\pm 1.19}$	$53.6_{\pm 2.40}$	$6.8_{\pm 0.78}$	$22.0_{\pm 1.30}$	$17.2_{\pm 0.82}$
Claude-Sonnet-4	$7.4_{\pm 0.98}$	$14.6_{\pm 2.12}$	$5.6_{\pm 0.89}$	$40.2_{\pm 1.41}$	$12.8_{\pm 1.29}$	$17.2_{\pm 1.39}$	$16.3_{\pm 0.81}$
Qwen3-235B-A22B	$14.0_{\pm 1.63}$	$3.8_{\pm 0.80}$	$5.6_{\pm 1.10}$	$31.2_{\pm 1.88}$	$8.6_{\pm 1.57}$	$18.4_{\pm 1.17}$	$13.6_{\pm 0.85}$
Qwen3-1.7B							
Base	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.0_{\pm 0.00}$	$0.3_{\pm 0.15}$	$0.0_{\pm 0.00}$	$0.4_{\pm 0.20}$	$0.1_{\pm 0.05}$
+ PPO	$0.6_{\pm 0.21}$	$0.7_{\pm 0.30}$	$0.8_{\pm 0.27}$	$2.2_{\pm 0.47}$	$0.6_{\pm 0.30}$	$2.6_{\pm 0.46}$	$1.3_{\pm 0.23}$
+ GRPO	$1.0_{\pm 0.41}$	$1.1_{\pm 0.44}$	$1.4_{\pm 0.48}$	$3.4_{\pm 0.86}$	$1.0_{\pm 0.31}$	$3.6_{\pm 0.71}$	$1.9_{\pm 0.35}$
+ PPO + **DFE**	$1.2_{\pm 0.46}$	$1.3_{\pm 0.34}$	$1.7_{\pm 0.39}$	$4.8_{\pm 0.56}$	$1.4_{\pm 0.40}$	$5.0_{\pm 1.22}$	$2.6_{\pm 0.39}$
+ GRPO + **DFE**	$\mathbf{1.8}_{\pm 0.56}$	$\mathbf{1.9}_{\pm 0.59}$	$\mathbf{2.4}_{\pm 0.59}$	$\mathbf{6.5}_{\pm 1.20}$	$\mathbf{2.0}_{\pm 0.66}$	$\mathbf{6.7}_{\pm 0.74}$	$\mathbf{3.6}_{\pm 0.41}$
Qwen3-4B							
Base	$0.4_{\pm 0.20}$	$0.5_{\pm 0.25}$	$0.5_{\pm 0.25}$	$1.6_{\pm 0.58}$	$0.5_{\pm 0.25}$	$1.6_{\pm 0.61}$	$0.9_{\pm 0.15}$
+ PPO	$2.4_{\pm 0.67}$	$2.6_{\pm 0.80}$	$3.5_{\pm 0.59}$	$9.5_{\pm 1.19}$	$2.5_{\pm 0.69}$	$10.0_{\pm 1.16}$	$5.1_{\pm 0.64}$
+ GRPO	$3.5_{\pm 0.89}$	$3.8_{\pm 1.11}$	$5.5_{\pm 1.03}$	$13.5_{\pm 1.56}$	$3.7_{\pm 0.76}$	$14.0_{\pm 1.09}$	$7.3_{\pm 0.54}$
+ PPO + **DFE**	$4.5_{\pm 0.96}$	$4.8_{\pm 0.76}$	$6.2_{\pm 1.22}$	$17.5_{\pm 1.86}$	$4.7_{\pm 0.83}$	$18.0_{\pm 1.27}$	$9.3_{\pm 0.97}$
+ GRPO + **DFE**	$\mathbf{6.2}_{\pm 0.92}$	$\mathbf{6.6}_{\pm 0.89}$	$\mathbf{8.5}_{\pm 1.40}$	$\mathbf{22.5}_{\pm 2.67}$	$\mathbf{6.4}_{\pm 1.16}$	$\mathbf{22.0}_{\pm 1.55}$	$\mathbf{12.0}_{\pm 0.78}$
Qwen3-8B							
Base	$0.7_{\pm 0.31}$	$0.9_{\pm 0.36}$	$0.9_{\pm 0.45}$	$2.4_{\pm 0.59}$	$0.8_{\pm 0.40}$	$3.0_{\pm 0.86}$	$1.4_{\pm 0.35}$
+ PPO	$4.1_{\pm 1.02}$	$3.7_{\pm 0.73}$	$5.8_{\pm 1.09}$	$13.3_{\pm 1.64}$	$3.9_{\pm 0.72}$	$14.5_{\pm 1.11}$	$7.5_{\pm 0.78}$
+ GRPO	$5.9_{\pm 1.24}$	$6.6_{\pm 0.96}$	$8.0_{\pm 1.08}$	$20.7_{\pm 1.56}$	$6.9_{\pm 1.07}$	$22.8_{\pm 1.77}$	$11.8_{\pm 0.96}$
+ PPO + **DFE**	$7.0_{\pm 0.89}$	$6.6_{\pm 1.09}$	$9.7_{\pm 1.10}$	$26.4_{\pm 2.01}$	$7.7_{\pm 1.10}$	$29.4_{\pm 1.52}$	$14.5_{\pm 0.91}$
+ GRPO + **DFE**	$\mathbf{8.5}_{\pm 1.10}$	$\mathbf{8.8}_{\pm 0.84}$	$\mathbf{15.1}_{\pm 1.50}$	$\mathbf{33.6}_{\pm 1.72}$	$\mathbf{10.7}_{\pm 1.20}$	$\mathbf{33.2}_{\pm 2.21}$	$\mathbf{18.3}_{\pm 0.51}$

Table 3: Manufactoria `Prep.` ablations. Left, intervention strength α. Right, pass@k with $\alpha = 0.1$.

α	pass@1	Δ
0.00 (vanilla)	1.6	—
0.05	4.0	+2.4
0.10	**4.3**	**+2.7**
0.20	4.0	+2.4
0.50	3.1	+1.5

Method	$k = 1$	$k = 4$	$k = 8$	$k = 16$
GRPO	1.6	1.9	2.1	2.2
GRPO + DFE	**4.3**	**4.9**	**5.2**	**5.4**
Δ	+2.7	+3.0	+3.1	+3.2

6.3 Ablation Study on α and Pass@k

We ablate two design choices on Medium Manufactoria `Prep.` with Qwen3-4B GRPO, a task where vanilla GRPO collects almost no reward and therefore offers a clean test of both questions. The α sweep in Table 3 (left) peaks at $\alpha = 0.1$, but the curve is flat across a wide band, and $\alpha = 0$ recovers the vanilla main-table value as a calibration check. The robustness in this band shows that DFE does not rely on careful tuning of α. The pass@k comparison in Table 3 (right) shows that the gap $\Delta =$ DFE $-$ vanilla does not close as k grows, and is slightly larger at the largest budget than at $k = 1$. The gain persists across the tested sampling budgets up to $k = 16$. Appendix J further compares training with higher temperature, more rollouts, and an entropy bonus, and reports sensitivity to intervention timing and other design choices.

6.4 Training Dynamics

Figure 3 compares PPO, GRPO, and DAPO (Yu et al., 2025) on the unseen `Has` sub-task after training on the Basic tier. We measure performance by the fraction of test cases passed. Across all three model sizes, DFE improves held-out performance earlier than vanilla RL. With GRPO, the 1.7B and 4B models reach vanilla GRPO's step-150 performance by step 30, while the 8B model does so by step 60. Thus, DFE reaches the same performance level with substantially fewer RL updates. This observation is consistent with Theorem 3. Under the stated local dynamics, increasing the singular value associated with a generalization direction shortens its emergence time when the spectral gain exceeds the residual term. The curves show the corresponding improvement in task performance. Appendix I provides the experimental details, and Appendix J reports a separate comparison of emergence times across paired modular multiplication runs.

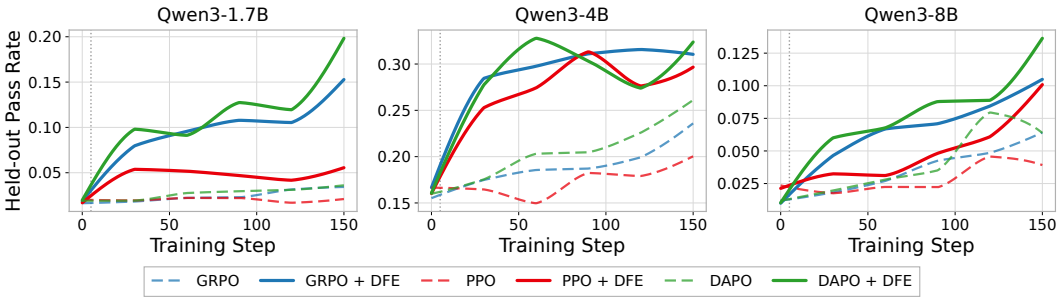


Figure 3: Transfer to unseen Manufactoria `Has`. Median test-case pass rate is plotted against RL updates.

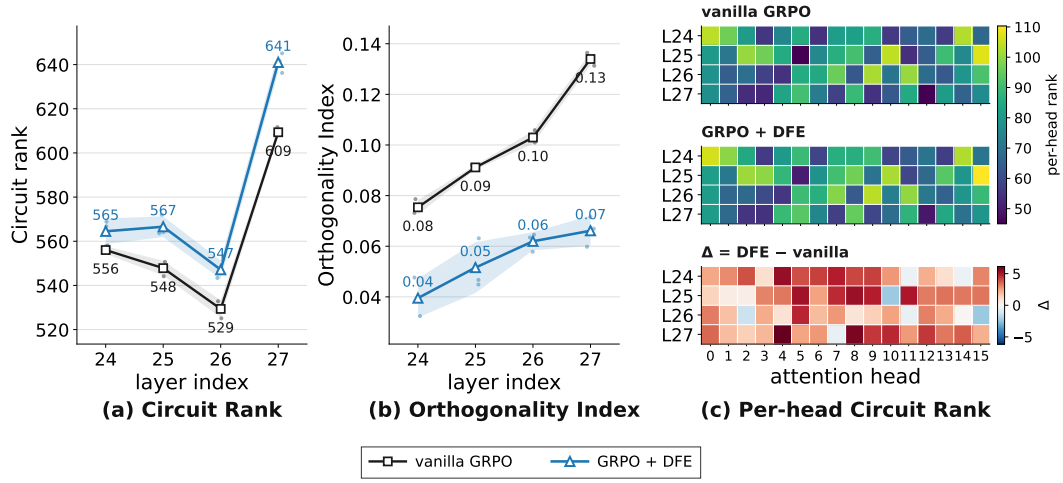


Figure 4: Circuit geometry in Qwen3-1.7B layers 24 to 27. (a) Circuit rank. (b) Orthogonality index. (c) Per-head rank and the change under DFE.

6.5 Visualization of Circuit Transition

Figure 4 compares the final four attention layers of Qwen3-1.7B trained with GRPO and GRPO + DFE. Panel (a) shows higher circuit rank under DFE at every layer, with the largest gap at layer 27. Panel (b) shows lower OI, indicating greater separation of task-token representations according to Definition 1. Panel (c) shows that the rank changes vary across heads, with both increases and decreases. Together, these observations associate the performance gains with changes in circuit geometry.

7 Conclusion

In this paper, we provide the first mechanistic characterization of delayed emergence in RLVR post-training, showing that it is governed by a phase transition from memorization to generalization characterized by changes in circuit rank, spectral concentration, and representation orthogonality. By recognizing the memorization plateau as the central obstacle to this transition and introducing a principled spectral intervention on the OV-circuit, our method DFE accelerates an aligned generalization coordinate under the stated local dynamics and significantly accelerates delayed emergence across baseline models and DELTA-CODE tasks, especially on fully out-of-distribution settings where the base model achieves nearly zero reward. Unlike prior work that characterizes the generalizing circuit only after the transition occurs, our approach analyzes circuit-level dynamics throughout RLVR training and translates this understanding into a closed-form method that directly accelerates task emergence. We anticipate this work will inspire broader exploration of mechanistic interventions in RLVR post-training, bridging the gap between circuit-level interpretability and task emergence in reinforcement learning.

AI USE STATEMENT

We used large language models only to assist with drafting, editing, and checking the mathematical exposition. These tools were not involved in the main ideation or implementation. This assistance included improving clarity, organizing explanations, and identifying inconsistencies in notation and presentation. The authors are responsible for the methods, results, and conclusions.

REPRODUCIBILITY STATEMENT

The theoretical assumptions and proofs are provided in Appendix C. Algorithm 1 specifies the intervention, and Appendix F describes the models, training setup, plateau detector, and evaluation protocol. Appendix J reports individual seed gains, emergence times, sensitivity analyses, and the computational cost of the intervention. The anonymous code is provided in the Supplementary Materials.

REFERENCES

David Demitri Africa, Sara M Kapoor, Theo Simon Sorg, and Challenger Mishra. Learning modular exponentiation with transformers. *arXiv preprint arXiv:2506.23679*, 2025.

Boaz Barak, Benjamin L Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang. Hidden Progress in Deep Learning: SGD Learns Parities Near the Computational Limit. *arXiv preprint arXiv:2207.08799*, 2022.

Peter Chen, Xiaopeng Li, Ziniu Li, Wotao Yin, Xi Chen, and Tianyi Lin. Exploration vs exploitation: Rethinking rlvr through clipping, entropy, and spurious reward. *arXiv preprint arXiv:2512.16912*, 2025.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. In *International Conference on Machine Learning*, pp. 10818–10838. PMLR, 2025.

DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948*, 2025.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Mingyuan Fan, Weiguang Han, Daixin Wang, Cen Chen, Zhiqiang Zhang, and Jun Zhou. When sharpening becomes collapse: Sampling bias and semantic coupling in RL with verifiable rewards. *arXiv preprint arXiv:2601.15609*, 2026.

Andrey Gromov. Grokking Modular Arithmetic. *arXiv preprint arXiv:2301.02679*, 2023.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations*, 2018.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. ProRL: Prolonged reinforcement learning expands reasoning boundaries in large language models. In *Advances in Neural Information Processing Systems*, volume 38, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. *Advances in Neural Information Processing Systems*, 35:34651–34663, 2022.

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

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress Measures for Grokking via Mechanistic Interpretability. In *International Conference on Learning Representations*, 2023.

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.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeffrey Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Isaac Bloom, et al. Open problems in mechanistic interpretability. *Transactions on Machine Learning Research*, 2025.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. Rl's razor: Why online reinforcement learning forgets less. *arXiv preprint arXiv:2509.04259*, 2025.

Yiyou Sun, Yuhan Cao, Pohao Huang, Haoyue Bai, Hannaneh Hajishirzi, Nouha Dziri, and Dawn Song. DELTA-code: How RL unlocks and transfers new programming algorithms in LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026.

Yuandong Tian. Provable Scaling Laws of Feature Emergence from Learning Dynamics of Grokking. *International Conference on Learning Representations (ICLR)*, 2026.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Chen Wang, Zhaochun Li, Jionghao Bai, Hexuan Deng, Ge Lan, and Yue Wang. Scope-rl: Stable and quantitative control of policy entropy in rl post-training. *arXiv preprint arXiv:2510.08141*, 2025.

Zhilin Wang, Yafu Li, Shunkai Zhang, Zhi Wang, Haoran Zhang, Xiaoye Qu, and Yu Cheng. New skills or sharper primitives? a probabilistic perspective on the emergence of reasoning in rlvr. *arXiv preprint arXiv:2602.08281*, 2026.

Yongzhong Xu. The geometry of multi-task grokking: Transverse instability, superposition, and weight decay phase structure. *arXiv preprint arXiv:2602.18523*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

APPENDIX

A LIMITATIONS AND BROADER IMPACTS

Limitations. This work studies delayed emergence in RLVR through the dynamics of Transformer OV circuits. We focus on the OV circuit because it is the most accountable component for tracing how internal token representations are routed into output-logit directions, making it a direct and measurable interface between circuit structure and model predictions. This focus does not mean other components are irrelevant, but it provides the clearest object for connecting mechanistic changes to the observed transition from memorization to generalization. Our theoretical analysis also separates the update directions into memorization, generalization, and noise subspaces. This type of decomposition is a standard abstraction in prior work on feature learning, grokking, and learning dynamics, and we use it to isolate the mechanism behind the memorization plateau and variance bottleneck. Empirically, we evaluate DFE on DELTA-Code with Qwen3 models under PPO and GRPO, where tasks have verifiable rewards and clear out-of-distribution structure. The current evidence most directly supports settings where a latent transferable rule exists and the verifier provides a reliable correctness signal. Extending the analysis to broader reasoning tasks, other architectures, and more complex verifier settings is an important direction for future work.

Broader Impacts. This work aims to make RLVR post-training more interpretable and more sample efficient by linking delayed generalization to measurable circuit-level changes. These diagnostics can help identify whether a model is only fitting training rollouts or learning a transferable computation, which may reduce wasted compute and make post-training behavior easier to audit. At the same time, any method that accelerates capability emergence should be used with care. If the verifier rewards an unintended behavior or contains exploitable shortcuts, DFE may also accelerate the wrong behavior. We therefore view verifier validation, held-out evaluation, and circuit-level monitoring as necessary safeguards when applying this type of intervention. More broadly, the paper suggests that mechanistic analysis can guide training-time interventions, but such interventions should be evaluated not only by benchmark gains but also by evidence about what computation the model has learned.

B NOTATIONS

Table 4 summarizes all the notations used throughout the paper. The notations in Section D are self-contained and not included in this table for brevity.

C PROOFS

C.1 THEORETICAL ASSUMPTIONS

Assumption 1 (Feature Subspaces)**.** The parameter space splits into three orthogonal subspaces: $\mathcal{M}$ (memorization directions that interpolate the batch), $\mathcal{G}$ (generalizable directions for the true target computation), and $\mathcal{R}$ (irrelevant noise directions). Each advantage-weighted feature decomposes as $A_i\phi_i = m_i + h_i + r_i$, with $m_i \in \mathcal{M}$, $h_i \in \mathcal{G}$, $r_i \in \mathcal{R}$, inducing $g = g_m + g_h + g_r$ where $g_m = \frac{1}{N}\sum_i m_i$ and similarly for g_h, g_r.

Assumption 2 (Gradient statistics)**.** Under Assumption 1, suppose the following conditions hold.

(A1) The memorization features $\{m_i\}_{i=1}^N$ are mutually independent with population mean $\mathbb{E}[m_i] = \bar{m}$, $\|\bar{m}\| = c_m > 0$, and bounded variance $\mathbb{E}\|m_i - \bar{m}\|^2 \le \sigma_m^2$.

(A2) The samples $\{r_i\}_{i=1}^N$ are mutually independent with $\mathbb{E}[r_i] = 0$ and $\mathbb{E}[\|r_i\|^2] \le C$.

(A3) Let $u \in \mathcal{G}$ be the unit vector along the true rule direction. The samples $\{h_i\}_{i=1}^N$ are mutually independent with $\mathbb{E}[\langle h_i, u\rangle] = \mu$ and $\mathrm{Var}(\langle h_i, u\rangle) = \sigma^2$. The component of h_i orthogonal to u has mean zero and total variance bounded by τ^2.

Assumption 3 (Local acceleration dynamics)**.** Consider a matched generalization coordinate of the original and edited OV circuits, with singular values $0 < s < s'$. Its local dynamics satisfy $\dot{z} = \eta s^2 z + r(t)$ and $\dot{z}' = \eta (s')^2 z' + r'(t)$, with the same effective learning rate $\eta > 0$, initial value

Table 4: Summary of notations used throughout the paper.

Symbol	Meaning
π_t	policy at training step t
$E_t \in \mathbb{R}^{n\times d},\ E_t^{\text{out}} \in \mathbb{R}^{d\times n}$	task-token input / output embedding matrices at step t
$W_{\text{OV},t} = W_V W_O$	OV-circuit weight at the targeted attention layer
$C_t = E_t\, W_{\text{OV},t}\, E_t^{\text{out}} \in \mathbb{R}^{n\times n}$	OV-circuit projection
$\rho(C_t)$	circuit rank: $\rho(W) = \exp\big(-\sum_i q_i \log q_i\big),\ q_i = \sigma_i^2 / \sum_j \sigma_j^2$
$\mathrm{OI}(t)$	orthogonality index $\|\hat{E}_t \hat{E}_t^\top - I_n\|_F / \sqrt{n(n-1)}$, with $\hat{E}_t$ row-normalized
$\mathcal{M},\ \mathcal{G},\ \mathcal{R}$	memorization, generalization, irrelevant-noise subspaces
$\mathcal{P} = \{1,\dots,k\}$	protected singular directions (top-k)
$\mathcal{B} = \{k{+}1,\dots,r\}$	candidate singular directions
$\sigma_i,\ \sigma_i'$	i-th spectral component of $W_{O,t}$ before / after DFE
$\bar{\sigma}_{\mathcal{P}}$	mean of $\{\sigma_i\}_{i\in\mathcal{P}}$
$g = g_m + g_h + g_r$	policy-gradient decomposition along $\mathcal{M},\ \mathcal{G},\ \mathcal{R}$
c_m	memorization mean magnitude, $\|\bar{m}\| = c_m > 0$
μ	per-sample alignment $\mathbb{E}[\langle h_i, u\rangle]$ with the true-rule direction $u \in \mathcal{G}$
N	rollout batch size used in the gradient estimate
$p_W(x)$	rollout success rate at prompt x, $p_W(x) = \mathbb{P}_{y\sim\pi(\cdot\|x)}\big[v(x,y)=1\big]$
$J(\pi_t)$	RLVR objective at policy π_t
$z,\ z'$	matched generalization coordinates before / after DFE
$s,\ s'$	generalization singular value before / after DFE
$T,\ T'$	hitting time to reach target coordinate magnitude ϵ
η	effective learning rate
ϵ	emergence threshold on the projection coordinate
$\alpha \in [0,\ 1]$	DFE intervention strength: $\sigma_i' = (1-\alpha)\sigma_i + \alpha\,\bar{\sigma}_{\mathcal{P}}$
k	size of the protected set $\mathcal{P}$

$z(0) = z'(0) = z_0 > 0$, and target $\epsilon > z_0$. Until their respective first hitting times T and T', the residuals satisfy $|r(t)| \le \delta z(t)$ and $|r'(t)| \le \delta z'(t)$ for a common $0 \le \delta < \eta s^2$.

C.2 Proof of Theorem 1

We give the formal version of Theorem 1 and its proof.

Theorem (Memorization Plateau, formal). Under Assumptions 1 and 2, we have

$$\mathbb{E}\|g_m\|^2 = c_m^2 + \mathcal{O}(1/N), \qquad \mathbb{E}\|g_h\|^2 = \mu^2 + \mathcal{O}(1/N), \qquad \mathbb{E}\|g_r\|^2 = \mathcal{O}(1/N).$$

Proof. By Assumption 1, the subspaces $\mathcal{M}, \mathcal{G}, \mathcal{R}$ form an orthogonal direct sum, so g_m, g_h, g_r are mutually orthogonal and $\|g\|^2 = \|g_m\|^2 + \|g_h\|^2 + \|g_r\|^2$. We bound each component.

Memorization component. By (A1), $\mathbb{E}[g_m] = \mathbb{E}[m_i] = \bar{m}$, and by independence

$$\mathbb{E}\|g_m - \bar{m}\|^2 = \frac{1}{N^2}\sum_{i=1}^{N} \mathbb{E}\|m_i - \bar{m}\|^2 \le \frac{\sigma_m^2}{N}.$$

Therefore $\mathbb{E}\|g_m\|^2 = \|\bar{m}\|^2 + \mathbb{E}\|g_m - \bar{m}\|^2 = c_m^2 + \mathcal{O}(1/N)$.

Generalization component. Project g_h onto u: $\langle g_h, u\rangle = \frac{1}{N}\sum_{i=1}^{N}\langle h_i, u\rangle$. By (A3) and independence, this average has mean μ and variance σ^2/N, so $\mathbb{E}[\langle g_h, u\rangle^2] = \mu^2 + \sigma^2/N$. The component of g_h orthogonal to u corresponds to candidate but incorrect generalization rules; by (A3) it has mean zero, and its squared norm has expectation at most τ^2/N. By Pythagoras,

$$\mathbb{E}\|g_h\|^2 = \mathbb{E}[\langle g_h, u\rangle^2] + \mathbb{E}\|g_h - \langle g_h, u\rangle u\|^2 = \mu^2 + \mathcal{O}(1/N).$$

Irrelevant component. By (A2), the samples $\{r_i\}$ have zero mean and bounded second moment, and by independence the cross terms vanish in expectation, so

$$\mathbb{E}\|g_r\|^2 = \frac{1}{N^2}\sum_{i=1}^{N}\mathbb{E}[\|r_i\|^2] \le \frac{C}{N}.$$

□

The three assumptions here capture distinct statistical roles of the three subspaces. Memorization features $\{m_i\}$ are assumed to share a common population mean $\bar{m}$ with $\|\bar{m}\| = c_m > 0$: each rollout contributes a consistent push along $\bar{m}$, reflecting the shared "fit-this-batch" signal that drives the policy to interpolate the training prompts. This common direction depends on the specific batch and does not transfer to unseen inputs, hence still lies in $\mathcal{M}$. Irrelevant features $\{r_i\}$ are zero-mean and independent across samples, so they contribute only sampling noise that averages out as N grows. Generalization features $\{h_i\}$ have a small per-sample alignment $\mu = \mathbb{E}[\langle h_i, u\rangle]$ with the true rule direction $u \in \mathcal{G}$: at initialization μ is small because several candidate rules are consistent with limited data, but it is the only component that points toward genuine generalization.

C.3 Proof of Theorem 2

Proof. By the score-function identity,

$$\nabla_W J(W) = \mathbb{E}_{x\sim\mathcal{D}}\,\mathbb{E}_{y\sim\pi(\cdot|x)}\big[v(x,y)\,\phi(x,y)\big],$$

where $\phi(x,y) = \nabla_W \log \pi(y \mid x)$. For the softmax policy,

$$\phi(x,y) = \nabla_W f_W(x,y) - \bar{\phi}(x), \qquad \bar{\phi}(x) := \mathbb{E}_{y'\sim\pi(\cdot|x)}[\nabla_W f_W(x,y')].$$

Conditioning on the value of $v(x,y) \in \{0,1\}$, define

$$\phi_1(x) := \mathbb{E}_{y\sim\pi(\cdot|x)}[\phi(x,y) \mid v=1], \quad \phi_0(x) := \mathbb{E}_{y\sim\pi(\cdot|x)}[\phi(x,y) \mid v=0],$$

so that $\Delta\phi(x) = \phi_1(x) - \phi_0(x)$ and the score-function mean property gives $p_W(x)\phi_1(x) + (1 - p_W(x))\phi_0(x) = 0$. The inner expectation evaluates to

$$\begin{aligned}
\mathbb{E}_y[v(x,y)\,\phi(x,y)] &= p_W(x)\,\phi_1(x) \\
&= p_W(x)\big[\phi_1(x) - p_W(x)\phi_1(x) - (1-p_W(x))\phi_0(x)\big] \\
&= p_W(x)\big(1-p_W(x)\big)\big(\phi_1(x) - \phi_0(x)\big) \\
&= p_W(x)\big(1-p_W(x)\big)\Delta\phi(x).
\end{aligned}$$

Taking the outer expectation over $x \sim \mathcal{D}$ yields the result. □

Scope of the reward analysis. Theorem 2 concerns the population gradient of expected binary reward on training prompts. For a graded reward $R \in [0,1]$, the corresponding promptwise gradient is $\mathbb{E}[(R - \mathbb{E}[R \mid x])\nabla_W \log \pi(y \mid x) \mid x]$, so learning depends on reward and feature covariance rather than the binary factor $p_W(x)(1-p_W(x))$. PPO clipping and GRPO or DAPO advantage normalization further change the update. Normalization can rescale an existing reward contrast, while a group with identical rewards has no within-group contrast. The theorem therefore isolates a source of weak learning signal rather than an exact update equation for these optimizers. Held-out performance measures transfer from this training process and does not contribute to its gradient.

C.4 Proof of Theorem 3

Proof. Write $a = \eta s^2$ and $b = \eta(s')^2$. The residual bounds imply

$$(a-\delta)z \le \dot{z} \le (a+\delta)z, \qquad (b-\delta)z' \le \dot{z}' \le (b+\delta)z'$$

until the respective hitting times. Since $b > a > \delta$ and both initial values equal $z_0 > 0$, the coordinates remain positive and increase toward ϵ.

Let $L = \log(\epsilon/z_0) > 0$. Integrating the logarithmic derivatives gives

$$\frac{L}{a+\delta} \le T \le \frac{L}{a-\delta}, \qquad \frac{L}{b+\delta} \le T' \le \frac{L}{b-\delta}.$$

Consequently,

$$\frac{T'}{T} \le \frac{a+\delta}{b-\delta}.$$

The condition $b-a > 2\delta$ makes this upper bound strictly smaller than one. When $\delta = 0$, the intervals reduce to $T = L/a$ and $T' = L/b$, giving

$$\frac{T'}{T} = \frac{s^2}{(s')^2}.$$

The bound is sharp under the stated assumptions. Equality is attained by $r(t) = \delta z(t)$ and $r'(t) = -\delta z'(t)$. □

Additive residuals. Suppose both trajectories instead satisfy a uniform additive bound $|r(t)|, |r'(t)| \le R$ until their respective hitting times. If $az_0 > R$, scalar comparison gives

$$z(t) \ge \frac{R}{a} + \left(z_0 - \frac{R}{a}\right) e^{at} \ge z_0.$$

The same argument applies to z' because $b > a$. Thus the relative residual bounds hold with $\delta = R/z_0$.

In particular, if the original $\mathcal{O}(p_W(x))$ remainder is bounded by $C\bar{p}$ uniformly along both trajectories, the theorem applies whenever

$$\eta s^2 z_0 > C\bar{p}, \qquad \eta\big((s')^2 - s^2\big) > \frac{2C\bar{p}}{z_0}.$$

These conditions account for both the residual magnitude and the initial coordinate. A small success probability alone does not justify discarding the remainder when the coordinate is also small.

Induced circuit amplification. The implemented intervention acts on W_O, while the theorem concerns the induced circuit $W_{\mathrm{OV}} = W_V W_O$. Write

$$W_O = \tilde{U}\Sigma\tilde{V}^\top$$

and hold W_V fixed during the edit. For each candidate direction $i \in \mathcal{B}$, define

$$c_i = \bar{\sigma}_{\mathcal{P}} - \sigma_i \ge 0, \qquad \sigma_i' = \sigma_i + \alpha c_i.$$

The protected components remain unchanged. Therefore,

$$\Delta W_O = W_O' - W_O = \alpha \sum_{i\in\mathcal{B}} c_i \tilde{u}_i \tilde{v}_i^\top,$$

and orthogonality of the singular bases gives

$$\tilde{U}_{\mathcal{P}}^\top \Delta W_O = 0, \qquad \Delta W_O \tilde{V}_{\mathcal{P}} = 0.$$

For a nonempty candidate set,

$$\|\Delta W_O\|_2 = \alpha \max_{i\in\mathcal{B}} c_i \le \alpha\bar{\sigma}_{\mathcal{P}}, \qquad \|\Delta W_{\mathrm{OV}}\|_2 \le \|W_V\|_2 \|\Delta W_O\|_2.$$

An empty candidate set gives $\Delta W_O = 0$.

The edit also controls the induced circuit energy. Define

$$d_i = (\sigma_i')^2 - \sigma_i^2 = 2\alpha\sigma_i c_i + \alpha^2 c_i^2 \ge 0.$$

Because reconstruction preserves the singular bases,

$$W_{\mathrm{OV}}'(W_{\mathrm{OV}}')^\top - W_{\mathrm{OV}} W_{\mathrm{OV}}^\top = \sum_{i\in\mathcal{B}} d_i (W_V\tilde{u}_i)(W_V\tilde{u}_i)^\top \succeq 0.$$

Every ordered singular value of the composed circuit is therefore nondecreasing. For any fixed unit direction q, the energy increase is

$$D(q) := \|q^\top W'_{\mathrm{OV}}\|_2^2 - \|q^\top W_{\mathrm{OV}}\|_2^2 = \sum_{i\in\mathcal{B}} d_i (q^\top W_V \tilde{u}_i)^2.$$

This increase is strict if $\alpha > 0$ and there exists a candidate i with $c_i > 0$ and $q^\top W_V \tilde{u}_i \neq 0$. Thus overlap with an amplified candidate component gives a sufficient condition for increasing energy along a fixed direction.

These guarantees concern the intervention with W_V fixed. The composed circuit's singular vectors may rotate, so strict amplification of a direction matched by overlap remains a separate condition. Preservation of the protected components also does not by itself guarantee preservation of policy reward.

Intervention strength. Suppose a unit generalization direction q is a left singular vector of both the original and edited OV circuits, with singular values $s > 0$ and s'. Then

$$(s')^2 - s^2 = D(q).$$

Define

$$A_q = \sum_{i\in\mathcal{B}} \sigma_i c_i (q^\top W_V \tilde{u}_i)^2, \qquad B_q = \sum_{i\in\mathcal{B}} c_i^2 (q^\top W_V \tilde{u}_i)^2.$$

It follows that

$$D(q) = 2\alpha A_q + \alpha^2 B_q.$$

Under the remaining assumptions of Theorem 3, strict acceleration follows if

$$\eta(2\alpha A_q + \alpha^2 B_q) > 2\delta.$$

For $B_q > 0$, this condition is equivalent to

$$\alpha > \frac{\sqrt{A_q^2 + 2B_q\delta/\eta} - A_q}{B_q}, \qquad \alpha \in [0, 1].$$

This gives an explicit intervention threshold when the aligned singular direction is retained across the edit. It does not require W_O and W_{OV} to share singular vectors.

D More Background on Transformer Circuits

Transformer circuits provide a linear-algebraic view of how a Transformer routes and transforms information through the residual stream (Elhage et al., 2021). Consider a Transformer with hidden dimension d and attention-head dimension d_h. Let $x_i^\ell \in \mathbb{R}^{1\times d}$ denote the residual representation of token position i at layer ℓ. For an attention head h in layer ℓ, let

$$W_Q^{(h,\ell)}, W_K^{(h,\ell)}, W_V^{(h,\ell)} \in \mathbb{R}^{d\times d_h}, \qquad W_O^{(h,\ell)} \in \mathbb{R}^{d_h\times d}.$$

The attention weight from query position i to source position j is

$$a_{ij}^{(h,\ell)} = \mathrm{softmax}_j\left(\frac{x_i^\ell W_Q^{(h,\ell)} \left(x_j^\ell W_K^{(h,\ell)}\right)^\top}{\sqrt{d_h}}\right),$$

where the softmax is taken over all source positions j. The output of this head at position i is

$$o_i^{(h,\ell)} = \sum_j a_{ij}^{(h,\ell)} x_j^\ell W_V^{(h,\ell)} W_O^{(h,\ell)} \in \mathbb{R}^{1\times d}.$$

This expression separates an attention head into two coupled circuits. The Query-Key circuit

$$W_{\mathrm{QK}}^{(h,\ell)} = W_Q^{(h,\ell)} W_K^{(h,\ell)\top} \in \mathbb{R}^{d\times d}$$

determines the attention pattern, namely which source tokens are read by each query token. The Output-Value circuit

$$W_{\mathrm{OV}}^{(h,\ell)} = W_V^{(h,\ell)} W_O^{(h,\ell)} \in \mathbb{R}^{d\times d}$$

determines what information is written from the attended source tokens back into the residual stream. Thus, the QK circuit controls routing, while the OV circuit controls the content and output effect of that routing. For a multi-head attention layer, the residual stream is updated as

$$x_i^{\ell+1} = x_i^{\ell} + \sum_{h=1}^{H} \sum_{j} a_{ij}^{(h,\ell)} x_j^{\ell} W_{\mathrm{OV}}^{(h,\ell)} + \mathrm{MLP}^{\ell}(x_i^{\ell}),$$

where H is the number of heads and $\mathrm{MLP}(\cdot)$ is the multi-layer perceptron. This form shows that the residual stream acts as a shared communication channel: each head reads from previous token representations and writes a structured update that can be used by later layers.

This decomposition is useful for studying algorithmic generalization because a model can obtain high training reward through different internal mechanisms. A memorizing solution may route instance-specific associations to the output, while a generalizing solution must organize token features into a reusable rule. These two solutions can have similar training reward but different circuit structure. Since the OV circuit writes attended information back into the residual stream, and the residual stream is later mapped to output logits, it provides a direct interface for tracing how task-token features affect prediction behavior.

To connect this circuit view to our analysis, let $\mathcal{V}_{\mathrm{task}}$ denote the set of n task tokens. At training step t, let $E_t \in \mathbb{R}^{n\times d}$ be the matrix whose rows are the input embeddings of these task tokens, and let $E_t^{\mathrm{out}} \in \mathbb{R}^{d\times n}$ be the output embedding matrix restricted to the same task-token set, with one output direction per task token. For a selected attention block, we write $W_{\mathrm{OV},t}$ for its OV circuit at step t. When the block contains multiple heads, $W_{\mathrm{OV},t}$ can denote either a specific head-level OV matrix or the aggregate OV contribution of the block, depending on the measurement used. We define the task-projected OV circuit as

$$C_t = E_t \cdot W_{\mathrm{OV},t} \cdot E_t^{\mathrm{out}} \in \mathbb{R}^{n\times n}.$$

The entry $(C_t)_{ab}$ measures how the OV circuit maps information from task token a toward the output-logit direction of task token b. This projection removes unrelated vocabulary directions and gives a compact circuit-level object for studying the task computation.

Our contribution is to use this circuit formalism to study the training trajectory itself. Prior circuit analyses mainly identify mechanisms after learning has occurred, such as induction heads or task-specific circuits in grokking. We instead track the spectrum and geometry of C_t during RLVR training and use it to characterize delayed emergence as a structural transition from memorization to generalization. This also gives a direct intervention target: DFE modifies the spectrum of W_O and thereby changes $W_{\mathrm{OV},t}$, strengthening weak candidate directions while preserving the dominant components of W_O. In this sense, the Transformer-circuit view is not only explanatory in our paper; it links circuit dynamics, RL gradient behavior, and a concrete method for accelerating emergence.

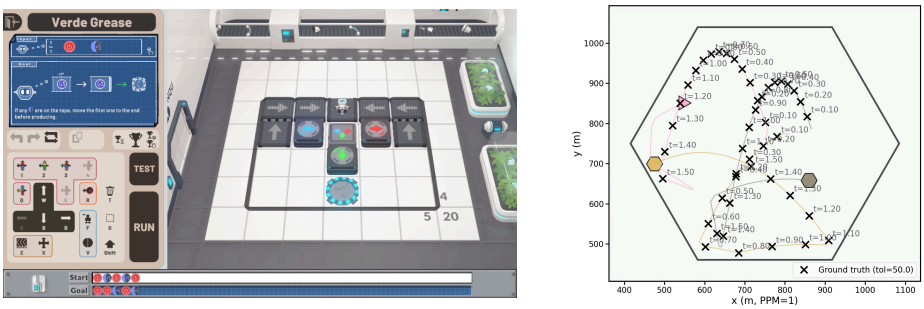
D.1 Interpreting the Circuit Diagnostics

Circuit rank in Definition 1 is an effective rank of the squared singular-value distribution. A lower value indicates greater spectral concentration, while a higher value indicates that energy is spread over more directions. We use spectral sparsity in this sense. The metric does not count active neurons or identify task-relevant directions. References to alive circuits describe directions carrying spectral energy, summarized by effective rank without a binary activation threshold.

The orthogonality index measures pairwise similarity of normalized task-token representations. For a representation matrix $R_{t,\ell}$ at layer ℓ, the corresponding quantity is

$$\mathrm{OI}_{\ell}(t) = \frac{\|\widehat{R}_{t,\ell}\widehat{R}_{t,\ell}^{\top} - I_n\|_F}{\sqrt{n(n-1)}}.$$

Its square is the mean squared cosine similarity between distinct rows. Layer comparisons require a fixed token set and a specified representation at each layer. Using only the shared input embedding E_t gives a single global index. These geometric measurements describe representation changes, while identifying generalizing computations requires additional functional evidence.



(a) Manufactoria (b) BouncingSim

Figure 5: (a) **Manufactoria** and (b) **BouncingSim** tasks from DELTA-Code (Sun et al., 2026). Both tasks require models to understand generalizable logic or physics rules from the benchmark environments.

E Directed Feature Emergence Algorithm

The schedule uses the plateau detector in Appendix F or the fixed early edit in Appendix I.

Algorithm 1 Directed Feature Emergence (DFE)

Input: Initial policy π_0, verifier v, spectral energy threshold γ, intervention strength α, training steps T, intervention schedule $\mathcal{S}$
Output: Optimized policy π_T at step T

for $t = 0$ **to** T **do**
  Sample completions $y \sim \pi_t(\cdot|x)$ and compute verifier rewards $v(x, y)$
  Update policy π_t using RLVR objective (e.g., PPO, GRPO)
  if the chosen schedule $\mathcal{S}$ triggers an edit **and** DFE not yet applied **then**
    for each target late-layer attention block **do**
      Extract output projection W_O from π_t
      Compute SVD: $W_O = \tilde{U}\Sigma\tilde{V}^\top$ where $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$ with $\sigma_1 \geq \cdots \geq \sigma_r$
      Find partition index $k = \min\left\{ j \;\middle|\; \frac{\sum_{i=1}^{j} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} \geq \gamma \right\}$
      Compute protected mean: $\bar{\sigma}_{\mathcal{P}} = \frac{1}{k}\sum_{i=1}^{k} \sigma_i$
      Initialize $\sigma'_i = \sigma_i$ for $i \in \{1, \ldots, r\}$
      for $i = k + 1$ **to** r **do**
        $\sigma'_i \leftarrow (1 - \alpha)\sigma_i + \alpha\bar{\sigma}_{\mathcal{P}}$
      end for
      Reconstruct $W'_O = \tilde{U}\Sigma'\tilde{V}^\top$ where $\Sigma' = \mathrm{diag}(\sigma'_1, \ldots, \sigma'_r)$
      Update π_t with intervened W'_O
    end for
  end if
end for
return π_T

F Implementation Details

Base models and benchmarks. In Table 1 and 2, the Qwen3-1.7B rows use the pretrained `Qwen/Qwen3-1.7B` checkpoint, the Qwen3-4B rows use `Qwen/Qwen3-4B`, and the Qwen3-8B rows use `Qwen/Qwen3-8B`, following the DELTA-Code reference checkpoint (Sun et al., 2026). The models are loaded in bfloat16 precision for rollouts and float32 precision for optimizer states. We disable the reasoning path so that all comparisons are evaluated on the direct-answer distribution. Figure 5 visualizes representative benchmark samples. DELTA-Code contains two challenging task

families that require models to infer generalizable logic and physics rules, making it a suitable testbed for fully out-of-distribution evaluation where the model must go beyond memorizing training instances and acquire new task-level computations.

Reinforcement learning setup. For the main-table experiments, PPO and GRPO optimize a verifiable-reward objective with $K = 16$ rollouts per prompt, learning rate 5×10^{-6}, temperature 1.0, and zero KL penalty, matching the DELTA-Code configuration (Sun et al., 2026). Manufactoria training follows a warm-up phase under the per-test pass-rate reward and a consolidation phase under a binary all-pass reward. BouncingSim training optimizes the per-test pass rate against the 50-pixel tolerance used by the benchmark verifier. These experiments use 8 NVIDIA A100 GPUs with 40 GB memory. Figure 3 uses the separate transfer protocol in Appendix I.

DFE specifics. In the main experimental tables, DFE is applied once after the memorization plateau is detected, to the `o_proj` weight of the final four transformer layers. The spectral partition index k is selected by an energy threshold $\gamma = 0.90$. We use Equation (4) with $\alpha = 0.1$. The intervention in Figure 3 uses a fixed step and the settings in Appendix I.

Plateau detection. For the modular multiplication experiments, let r_t be the logged mean verifier reward of the rollout batch. Starting at $t = 5{,}000$, the detector triggers when the latest three logged rewards are all at least 0.95 and $|r_t - r_{t^-}| \leq 0.005$, where t^- is the latest log at or before $t - 5{,}000$. Logs are 200 training steps apart. This rule uses training reward only. The timing ablation in Appendix J varies the intervention around this trigger. Figure 3 instead uses a fixed intervention at step 5.

Evaluation protocol. All cells in Tables 1 and 2 are evaluated on the official DELTA-Code test splits (Sun et al., 2026), with $K = 16$ rollouts per prompt at temperature 1.0. A rollout is counted as a pass when it satisfies every ground-truth assertion from the provided test cases. For Manufactoria, the verifier simulates the DSL program against all input tapes. For BouncingSim, a Box2D physical simulation suite checks that the predicted positions fall within the benchmark's 50-pixel tolerance at every queried timestamp. Oracle rows (GPT-5, o4-mini, Claude-Sonnet-4, Qwen3-235B-A22B) are reproduced from DELTA-Code (Sun et al., 2026), and the remaining rows are measured by us under the same protocol.

Manufactoria training coverage. DELTA-Code releases public training splits for nine of the twelve Manufactoria sub-tasks: a Basic-Mix split covering the three Basic families (`App`, `Exa`, `Sta`), and per-task splits for `Has`, `Regex`, `Ends`, `Compr`, `Prep`, and `MnMx`. The remaining three sub-tasks (`BitOp`, `Symm`, `Add`) are evaluation-only based on the released source at https://huggingface.co/manufactoria, so the trained-model entries in those columns of Table 1 reflect zero-shot transfer learning capability.

Infrastructure. Inference is served by vLLM in bfloat16 precision with two-way tensor parallelism per worker on 8 NVIDIA A100 40 GB GPUs. Our trained-model rows in Tables 1 and 2 report means and standard deviations over three seeds $\{0, 1, 2\}$ at the end of training.

G BouncingSim: Per-Family Breakdown

Table 5 resolves the per-family results of Table 2 into the underlying difficulty tiers, for the four oracle baselines and the two strongest trained variants, Qwen3-4B with GRPO and Qwen3-4B with GRPO+DFE.

H Prompt Engineering

We follow the prompt templates released with DELTA-Code (Sun et al., 2026), both during RL training and at evaluation. Each row of the released dataset contains a self-contained `messages` list (system + user for Manufactoria; a single user message for BouncingSim) that already specifies the task semantics; we feed this list to the tokenizer through its standard chat template and never inject

Table 5: BouncingSim pass@1 (%) per physics family and difficulty tier. Oracle rows are reproduced from DELTA-Code (Sun et al., 2026).

Family	Model	Basic	Easy	Medium	Hard	Extreme
RotObj	GPT-5	52.0	11.0	5.0	7.0	0.0
	o4-mini	15.0	5.0	8.0	3.0	0.0
	Claude-Sonnet-4	23.0	7.0	5.0	2.0	0.0
	Qwen3-235B-A22B	59.0	6.0	5.0	0.0	0.0
	Qwen3-4B + GRPO	9.0	3.5	3.5	1.0	0.5
	Qwen3-4B + GRPO + DFE	16.0	7.5	5.5	1.5	0.5
RotBox	GPT-5	61.0	47.0	31.0	23.0	12.0
	o4-mini	18.0	7.0	5.0	5.0	3.0
	Claude-Sonnet-4	32.0	18.0	14.0	9.0	0.0
	Qwen3-235B-A22B	9.0	5.0	5.0	0.0	0.0
	Qwen3-4B + GRPO	8.5	4.5	4.0	1.5	0.5
	Qwen3-4B + GRPO + DFE	14.5	8.5	7.0	2.0	1.0
MovBox	GPT-5	76.0	64.0	58.0	0.0	0.0
	o4-mini	18.0	9.0	8.0	0.0	0.0
	Claude-Sonnet-4	23.0	5.0	0.0	0.0	0.0
	Qwen3-235B-A22B	23.0	5.0	0.0	0.0	0.0
	Qwen3-4B + GRPO	13.0	7.5	4.5	2.0	0.5
	Qwen3-4B + GRPO + DFE	18.5	12.5	8.0	3.0	0.5
Gravity	GPT-5	85.0	83.0	81.0	76.0	69.0
	o4-mini	64.0	57.0	56.0	52.0	39.0
	Claude-Sonnet-4	45.0	50.0	39.0	35.0	32.0
	Qwen3-235B-A22B	45.0	27.0	34.0	32.0	18.0
	Qwen3-4B + GRPO	24.0	17.5	18.0	6.0	2.0
	Qwen3-4B + GRPO + DFE	40.0	27.5	28.0	12.0	5.0
MultiBox	GPT-5	32.0	16.0	0.0	1.0	0.0
	o4-mini	22.0	12.0	0.0	0.0	0.0
	Claude-Sonnet-4	23.0	18.0	14.0	9.0	0.0
	Qwen3-235B-A22B	23.0	5.0	0.0	10.0	5.0
	Qwen3-4B + GRPO	8.0	4.5	2.5	3.0	0.5
	Qwen3-4B + GRPO + DFE	14.0	8.0	4.5	4.5	1.0
MultiObj	GPT-5	78.0	52.0	11.0	7.0	0.0
	o4-mini	65.0	32.0	7.0	6.0	0.0
	Claude-Sonnet-4	41.0	36.0	9.0	0.0	0.0
	Qwen3-235B-A22B	59.0	33.0	0.0	0.0	0.0
	Qwen3-4B + GRPO	30.0	19.5	13.5	5.0	2.0
	Qwen3-4B + GRPO + DFE	45.0	31.0	22.0	9.0	3.0

an additional system prompt. This ensures every entry in Tables 1 and 2 is directly comparable to the released DELTA-Code baselines.

H.1 MANUFACTORIA

Manufactoria prompts pair a condensed DSL specification in the system message with a per-instance task description in the user message. The model must emit a single triple-backtick block containing a complete DSL program; the verifier executes the program against ground-truth tape inputs.

Manufactoria: prompt template

System. You are a Manufactoria DSL programmer. Given a task, write a DSL program that processes tapes of colored tokens (`R`, `B`, `Y`, `G`) and either accepts or rejects each input.
Primitives. `START` (entry node), `END` (accept), `PULLER_RB`/`PULLER_YG` (consume the front token and route on its color, with optional `[EMPTY]` branch), `PAINTER_<C>` (append color C to the back of the tape and continue).
Routing. Any unhandled or `NONE` branch immediately rejects; execution is bounded to 1000 steps; a tape is accepted iff execution reaches an `END` node. Output only the DSL program, wrapped in a single triple-backtick block.

User. *(per-instance task description, e.g. `Has-family.`)* Accept exactly the tapes that contain the substring `RB`; reject everything else. Examples: `RBR` accepts; `BR` rejects; `RR` rejects; `RRBB` accepts.

H.2 BouncingSim

BouncingSim prompts are single-turn and present a scene specification (containers, balls, physics) together with the closed-form mechanics the model must use. The model implements a single Python function `predict_position(t)` that returns the 2D position of every ball at an arbitrary future time t; the verifier compares predictions against a reference Box2D simulation under a 50-pixel tolerance.

BouncingSim: prompt template

User. *Polygon Dynamics Prediction.* Implement `predict_position(t)` that returns the $[x, y]$ positions of all balls at an arbitrary future time t.
Mechanics. Use closed-form constant-acceleration kinematics $x(t) = x_0 + v_{x_0}\, t + \frac{1}{2} a_x\, t^2$, perfectly elastic boundary reflections via $v' = v - 2\,(v \cdot \hat{n})\,\hat{n}$, and finite-size polygon contact for both balls and containers. Positions are in meters, time in seconds.
Scene specification (excerpt). Container: regular polygon, 4 sides, radius 225.0 m, center $(750, 750)$. Ball: regular polygon, 3 sides, radius 40.0 m, initial position $(780, 685)$, initial velocity $(173.83, -297.07)$ m/s. Acceleration $(0, -300)$ m/s^2.
Output a single Python code block; rendering, animation, and explanatory text are out of scope.

I Training Dynamics Protocol

Task and metric. Figure 3 evaluates transfer from the Manufactoria Basic tier to the unseen `Has` sub-task. Training uses only `basic_mix_train.jsonl`, and evaluation uses `data/has_test.jsonl`. Dense pass rate averages the fraction of test cases passed by each generated program, first over samples and then over evaluation prompts. It measures partial correctness and differs from binary pass@1 in Table 1.

Updates and evaluation. Each RL update samples eight completions for each of eight prompts, giving 64 rollouts with at most 512 generated tokens each. The 150-step budget contains 9,600 rollouts. Evaluation occurs at steps $0, 30, 60, 90, 120, 150$, using 20 held-out prompts with eight samples each at temperature 1.0. Each plotted point is the median across seeds. Each GRPO arm uses 3 seeds at 1.7B, 4B and 8B. Each PPO and DAPO arm also uses 3 seeds at every scale. No dispersion band is shown. Lines blend a shape-preserving cubic interpolant and linear interpolation with weights 0.55 and 0.45 solely for display.

Intervention and checkpoint comparisons. The vertical dotted line marks a single edit at step 5, with $\alpha = 0.4, 0.6, 0.4$ at 1.7B, 4B, and 8B, respectively, $\gamma = 0.90$, and the final 16 attention output projections as targets. Before the edit, paired arms share the same configuration but have separate sampling randomness. The GRPO comparisons in Section 6.4 use the measured medians at steps 30, 30, and 60 for DFE and step 150 for vanilla GRPO, rather than crossings of the interpolated curves. The main-table intervention settings and modular multiplication detector are specified in Appendix F.

J Additional Experimental Results

Variation across training seeds. Table 6 reports the paired DFE-minus-vanilla gain across three training seeds. Each gain averages the 12 Manufactoria sub-tasks or six BouncingSim families equally. The paired runs use the same seed and are identical before the intervention. All 24 seed differences are positive. We report the mean and sample SD of the paired gains, with nominal 95% t intervals.

Table 6: DFE gains in pass@1, in percentage points, across three seeds. SD and intervals describe the paired gain.

Benchmark	Model	Optimizer	Gain	95% interval
Manufactoria	Qwen3-1.7B	PPO	4.95 ± 0.22	$[4.41, 5.49]$
	Qwen3-1.7B	GRPO	6.62 ± 0.34	$[5.78, 7.46]$
	Qwen3-4B	PPO	11.78 ± 0.52	$[10.50, 13.06]$
	Qwen3-4B	GRPO	15.18 ± 0.63	$[13.62, 16.74]$
BouncingSim	Qwen3-1.7B	PPO	1.32 ± 0.13	$[0.99, 1.65]$
	Qwen3-1.7B	GRPO	1.63 ± 0.15	$[1.26, 2.00]$
	Qwen3-4B	PPO	4.20 ± 0.26	$[3.55, 4.85]$
	Qwen3-4B	GRPO	4.70 ± 0.31	$[3.93, 5.47]$

Individual seed gains. Table 7 gives the DFE-minus-vanilla benchmark-average pass@1 gain for every training seed. Manufactoria averages its 12 sub-tasks equally, and BouncingSim averages its six physics families equally. For the three paired differences, the nominal 95% interval is $\bar{\Delta} \pm 4.303 s_\Delta / \sqrt{3}$, where s_Δ is their sample SD.

Table 7: Individual paired DFE gains in final pass@1, in percentage points.

Benchmark	Model	Optimizer	Seed 0	Seed 1	Seed 2
Manufactoria	Qwen3-1.7B	PPO	4.70	5.10	5.05
	Qwen3-1.7B	GRPO	6.25	6.70	6.91
	Qwen3-4B	PPO	11.20	11.95	12.19
	Qwen3-4B	GRPO	14.50	15.30	15.74
BouncingSim	Qwen3-1.7B	PPO	1.18	1.34	1.44
	Qwen3-1.7B	GRPO	1.50	1.60	1.79
	Qwen3-4B	PPO	3.95	4.18	4.47
	Qwen3-4B	GRPO	4.38	4.72	5.00

Individual emergence times. Table 8 reports the detector step t_d, sustained OOD hitting time t_{50}, and paired reduction $\Delta t_{50} = t_{50}^{\text{van}} - t_{50}^{\text{DFE}}$ for all ten modular multiplication comparisons. We define t_{50} as the first of five consecutive OOD evaluations at or above 50%, with evaluations every 200 training steps. PPO seed 1 is right-censored at 140k steps. The detector precedes both observed hitting times in every uncensored comparison.

Exploration comparisons. We compare exploration settings on Qwen3-4B GRPO and Manufactoria `Prep.` over three seeds. All conditions use the same optimizer updates and maximum sampled tokens per update. The 32-rollout conditions use fewer distinct prompts per update. Table 9 reports final pass@1 as mean and sample SD. The vanilla and default DFE reference results in Tables 9 and 11 use the corresponding values from Table 1.

Intervention timing. We compare early, detector-timed, late, and repeated edits on modular multiplication with PPO seeds 0, 2, and 4. All schedules use the same training horizon within this ablation. Table 10 reports mean and sample SD.

Design choices. Table 11 varies target layers, projection, γ, α, and the number of edits on Qwen3-4B GRPO and Manufactoria `Prep.`. Results are mean and sample SD over three seeds.

Table 8: Modular multiplication timing for every seed, in thousands of training steps. The $>$ symbol marks right censoring.

Optimizer	Seed	t_d	Vanilla t_{50}	DFE t_{50}	Δt_{50}
PPO	0	22.8	45.0	31.4	13.6
	1	20.0	> 140	64.2	> 75.8
	2	18.6	38.4	27.6	10.8
	3	22.8	112.6	54.8	57.8
	4	19.4	58.2	34.6	23.6
GRPO	0	26.4	36.2	30.8	5.4
	1	27.6	45.2	33.4	11.8
	2	25.8	47.0	32.6	14.4
	3	25.2	80.8	45.6	35.2
	4	22.6	51.2	31.0	20.2

Table 9: Exploration comparisons on Qwen3-4B GRPO and Manufactoria `Prep..` Final pass@1 (%) is mean $\pm$ sample SD over three seeds.

Condition	Temperature	Rollouts per prompt	Pass@1
Vanilla	1.0	16	1.6 ± 0.50
Higher temperature	1.2	16	2.0 ± 0.2
More rollouts	1.0	32	2.4 ± 0.3
Entropy coefficient 0.01	1.0	16	2.2 ± 0.2
DFE	1.0	16	4.3 ± 0.89
DFE with more rollouts	1.0	32	4.8 ± 0.3

Table 10: Timing ablation on modular multiplication with PPO seeds 0, 2, and 4. Times are in thousands of training steps.

Schedule	Intervention step	t_{50}	Final OOD accuracy (%)
No DFE	None	47.2 ± 10.1	68.4 ± 3.6
Early DFE	$t_d - 5$k	35.9 ± 3.6	75.4 ± 1.3
Plateau DFE	t_d	31.2 ± 3.5	78.1 ± 0.9
Late DFE	$t_d + 5$k	39.0 ± 3.7	74.1 ± 1.2
Repeated DFE	t_d and $t_d + 5$k	31.0 ± 3.5	78.0 ± 1.1

Table 11: DFE design choices on Qwen3-4B GRPO and Manufactoria `Prep..` Results follow the listed setting order.

Factor	Settings	Final pass@1 (%)
Target layers	Final 2 / final 4 / final 8	3.8 ± 0.3 / 4.3 ± 0.89 / 4.1 ± 0.4
Projection	`o_proj` / `v_proj` / `q_proj`	4.3 ± 0.89 / 3.1 ± 0.3 / 2.4 ± 0.2
γ	0.80 / 0.90 / 0.95	3.9 ± 0.3 / 4.3 ± 0.89 / 4.1 ± 0.3
α	0.05 / 0.10 / 0.20	4.0 ± 0.3 / 4.3 ± 0.89 / 4.0 ± 0.4
Schedule	One edit / repeated edit	4.3 ± 0.89 / 4.2 ± 0.3

Computational overhead. Table 12 reports the one-time cost of editing four target W_O matrices. Values are mean and sample SD over ten repetitions on one NVIDIA B200 after two warmups. Timing includes tensor input and output, the spectral edit, SVD, reconstruction, transfers, and diagnostics. The memory measurement is the peak allocator increase. The Qwen3.5-9B result measures editing cost only.

Table 12: One-time editing cost for four target W_O matrices.

Model	Target layers	Wall time (s)	Peak increase (MiB)
Qwen3-1.7B	24 to 27	1.536 ± 0.013	171.9
Qwen3-4B	32 to 35	2.715 ± 0.033	411.7
Qwen3.5-9B	Attention layers 19, 23, 27, 31	5.247 ± 0.047	681.2