# DIMENSION IS FREE, HORIZON IS NOT: WHEN EVOLUTION STRATEGIES RIVAL POLICY GRADIENTS FOR LLM FINE-TUNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Policy gradient (PG) methods such as GRPO are the dominant approach to LLM fine-tuning, but they suffer from imprecise credit assignment, a limitation that becomes especially pronounced in long-horizon agentic tasks. Recent work has established Evolution Strategies (ES) as a scalable alternative to PG (Qiu et al., 2026). Because ES evaluates each perturbed model by its total return, it avoids per-token credit assignment, which makes it a natural candidate for long-horizon settings. We analyze ES and PG within a single framework, as two estimators of the same gradient that explore in different currencies. ES perturbs parameters and therefore pays a dimension tax, whereas PG samples actions and therefore pays a horizon tax. We show that the taxes actually incurred are effective quantities. For ES, the tax is $D_{\text{eff}}$, the number of reward-relevant parameter directions. For PG, it is $T_{\text{eff}}$, the number of token positions that contribute reward-relevant variance, which increases only when reward-irrelevant tokens share both credit and weights with reward-carrying ones. To disentangle the two, we construct a synthetic task in which $D_{\text{eff}}$ and $T_{\text{eff}}$ can be controlled independently. On this task, PG outperforms ES at high $D_{\text{eff}}$ and short $T_{\text{eff}}$, whereas ES outperforms PG when $T_{\text{eff}}$ is long relative to $D_{\text{eff}}$, and this region expands with the training budget of ES. We then estimate proxies for $D_{\text{eff}}$ and $T_{\text{eff}}$ on 27 NLP, mathematical reasoning, and agentic tasks with Qwen2.5-7B-Instruct and relate the resulting atlas to head-to-head ES–PG comparisons. Our results support ES as a scalable alternative for LLM fine-tuning, particularly for long-horizon tasks with sparse credit.

## 1 INTRODUCTION

Policy gradient (PG) methods such as GRPO dominate reinforcement-learning fine-tuning of language models (Shao et al., 2024; Guo et al., 2025). Nonetheless, they are known to struggle with credit assignment, and long-horizon agentic tasks such as Terminal-Bench make this problem especially pronounced. Tmax reports that GRPO training becomes more unstable as the number of agent turns grows, and only succeeds after replacing GRPO with a variant that masks tokens with mismatched training and inference probabilities, reducing that mismatch, and enlarging the rollout groups (Ivison et al., 2026). T1 relies on stabilized actor–critic training and rewards for partial task completion (Yang et al., 2026), and TRACE assigns credit to individual tool actions (Tao et al., 2026). Taken together, these works show that training PG successfully on long horizons requires considerable additional engineering. This difficulty is rooted in how PG assigns credit. Under terminal credit, every generated token enters the PG gradient, so even tokens that do not affect the reward add variance to it.

Evolution Strategies (ES) follow a different principle. ES perturbs the model's parameters and scores each perturbed model by its total return (Salimans et al., 2017; Sehnke et al., 2010), and therefore does not need to assign credit to individual tokens. Recent work has shown that ES scales to LLM fine-tuning (Qiu et al., 2026). These observations lead us to ask whether ES can be applied to long-horizon fine-tuning, and whether it could have an advantage over PG there. In principle, the horizon problem that affects PG is largely absent for ES. When reward-irrelevant actions leave the return of each perturbed model unchanged, they also leave the ES estimate unchanged, whereas under terminal credit the same actions still add noise to the PG estimate.

However, ES pays a cost of its own. The two methods explore in different currencies. ES explores in parameter space and pays for dimension, while PG explores in action space and pays for horizon (Vemula et al., 2019). For an LLM with billions of parameters, the dimension cost would seem



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.

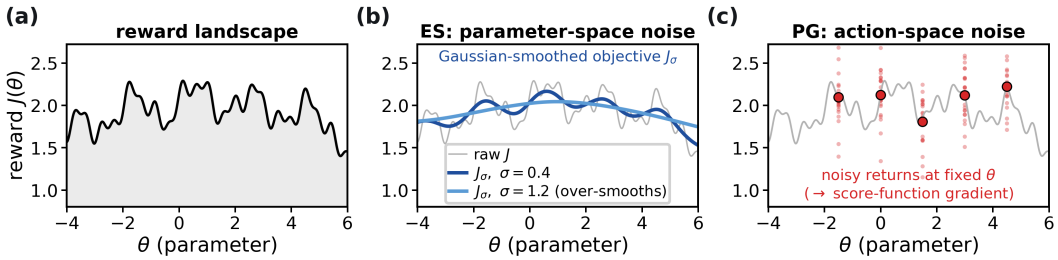


Figure 1: **Parameter-space and action-space exploration. (a)** The expected reward landscape $J(\theta)$. **(b)** ES perturbs the parameters and estimates the gradient of the smoothed objective $J_\sigma$. **(c)** PG samples actions and estimates the gradient of $J$. The ES target gradient approaches the PG target gradient as $\sigma \to 0$.

prohibitive. In practice, only the variance along reward-relevant directions affects learning, and fine-tuning is known to have low intrinsic dimension (Aghajanyan et al., 2021). Indeed, Liang et al. (2026) show that ES scales because its improvement is dominated by a small number of parameter directions. We therefore describe the two costs by an effective dimension $D_{\text{eff}}$, the number of reward-relevant parameter directions, and an effective horizon $T_{\text{eff}}$, the number of token positions that contribute reward-relevant PG variance. Unlike the sequence length, $T_{\text{eff}}$ counts a reward-irrelevant token only when it receives credit from the reward and its gradient overlaps the reward-relevant directions. In an autoregressive LLM this overlap is expected, because the same weights generate every token.

Because longer tasks also tend to be harder, the two costs are difficult to separate in real models. To study them independently, we design a synthetic task in which a policy generates $k$ reward-carrying tokens and $F$ filler tokens. The number of targets, the filler length, the parameter count, and weight sharing can each be varied on their own, which allows us to set $D_{\text{eff}}$ and $T_{\text{eff}}$ separately. Training both methods on this task at a matched budget, we find that PG performs better at high $D_{\text{eff}}$ and short $T_{\text{eff}}$, while ES performs better when $T_{\text{eff}}$ is long relative to $D_{\text{eff}}$. The region where ES performs better expands with its training budget, and the crossover also depends on the policy class (§4).

Finally, we examine where real tasks lie. We measure proxies for $D_{\text{eff}}$ and $T_{\text{eff}}$ on 27 NLP, mathematical reasoning, and agentic tasks with Qwen2.5-7B-Instruct, and compare the resulting atlas with head-to-head ES–PG training results and with comparisons reported in prior work.

**Contributions.** We make the following contributions:

- **A unified account of the two exploration costs.** We treat ES and PG as two estimators of one gradient and show that they pay $D_{\text{eff}}$ and $T_{\text{eff}}$, with $T_{\text{eff}}$ gated by credit and weight sharing (§2, §3).
- **A controlled testbed that separates the two costs.** A token-generation task in which $D_{\text{eff}}$ and $T_{\text{eff}}$ can be set independently, isolating how reward-irrelevant tokens affect PG (§2.3).
- **A trained comparison across the plane.** We map where ES outperforms PG across the synthetic grid, and how the crossover moves with training budget and policy class (§4).
- **A measured task atlas.** We estimate proxies for $D_{\text{eff}}$ and $T_{\text{eff}}$ on 27 NLP, math, and agentic tasks with Qwen2.5-7B-Instruct and compare them with head-to-head ES–PG results (§5).

Together, these results suggest that ES offers a scalable alternative to PG for long-horizon fine-tuning, particularly for tasks with low $D_{\text{eff}}$ and long $T_{\text{eff}}$.

## 2 Two Exploration Currencies: Parameters and Actions

We first formalize the gradient estimates of the two methods and then measure their variance in a controlled token-generation task. We optimize expected return $J(\theta) = \mathbb{E}_{a\sim\pi_\theta}[R(a)]$, with gradient $g^\star = \nabla_\theta J(\theta)$.

**Where exploration enters (Fig. 1).** ES samples perturbed parameters $\theta' \sim \mathcal{N}(\theta, \sigma^2 I)$ and estimates the gradient of the Gaussian-smoothed objective $J_\sigma$ (Fig. 1b). PG holds the parameters fixed, samples actions $a \sim \pi_\theta$, and weights the gradients of their log probabilities by the observed return (Fig. 1c). The ES target gradient approaches the PG target gradient as $\sigma \to 0$.

### 2.1 GRADIENT ESTIMATES IN THE TWO EXPLORATION SPACES

Both constructions follow the likelihood-ratio identity applied to their respective sampling distributions.

**Evolution strategies (parameter space).** ES (Salimans et al., 2017) places an isotropic Gaussian on the parameters and optimizes $J_\sigma(\theta) = \mathbb{E}_\varepsilon[J(\theta + \sigma\varepsilon)]$, with

$$\hat{g}_{\text{ES}} = \tfrac{1}{\sigma}\, R\big(a; \theta + \sigma\varepsilon\big)\, \varepsilon, \qquad \mathbb{E}[\hat{g}_{\text{ES}}] = \nabla_\theta J_\sigma \to g^\star \;\; (\sigma \to 0), \tag{1}$$

which is REINFORCE applied to the distribution over parameters.

**Policy gradient (action space).** PG (Williams, 1992) uses the gradient of each sampled action's log probability. For a single length-$T$ trajectory $a \sim \pi_\theta$,

$$\hat{g}_{\text{PG}} = \textstyle\sum_{t=1}^{T} \hat{A}_t\, \psi_t, \qquad \mathbb{E}_{a\sim\pi_\theta}[\hat{g}_{\text{PG}}] = g^\star, \qquad \psi_t \equiv \nabla_\theta \log \pi_\theta(a_t \mid s_t), \tag{2}$$

where $\hat{A}_t$ is the advantage assigned to token $t$. With terminal credit, $\hat{A}_t = R(a) - b$, where an action-independent baseline $b$ leaves the expected gradient unchanged. In practice the baseline is the mean reward of the $G$ rollouts in each update. It scales the expected gradient by $1 - 1/G$, which does not affect our scaling argument. More localized credit can reduce the contribution of irrelevant tokens. In the likelihood-ratio formulation, the log-density gradients are $\varepsilon/\sigma$ for the parameter perturbation and $\psi_t$ for each sampled action. We refer to these log-density gradients as *scores* below.

### 2.2 THE NAIVE DIMENSION AND HORIZON CURSES

We next compare how the total variance of each estimator depends on parameter count and trajectory length, before restricting attention to reward-relevant directions. At a fixed $\theta$ and matched per-update budget, and under the covariance assumptions of App. A, the two variances scale as

$$\boxed{\operatorname{tr}\operatorname{Cov}\big(\hat{g}_{\text{ES}}\big) \;\propto\; D, \qquad \operatorname{tr}\operatorname{Cov}\big(\hat{g}_{\text{PG}}\big) \;\propto\; T \cdot \kappa.} \tag{3}$$

Here $\kappa$ is the per-step *score energy*,

$$\kappa \;\equiv\; \mathbb{E}_{a_t\sim\pi}\big\|u_t\big\|^2, \qquad u_t \equiv \nabla_{z_t} \log \pi(a_t \mid z_t), \tag{4}$$

the expected squared norm of the score with respect to the action distribution's own parameters $z_t$ (the logits of a categorical policy, the mean of a Gaussian one). The parameter-space score of Eq. (2) is $\psi_t = J_t^\top u_t$ with readout Jacobian $J_t = \partial z_t/\partial\theta$, whose bounded magnitude we absorb into the proportionality constant.

Every ES estimate has the form $\hat{g}_{\text{ES}} = \Delta\,\varepsilon$ with $\varepsilon \sim \mathcal{N}(0, I_D)$, where $\Delta$ is a scalar coefficient computed from the returns. It equals $R(\theta+\sigma\varepsilon)/\sigma$ in Eq. (1) and $[R(\theta+\sigma\varepsilon) - R(\theta-\sigma\varepsilon)]/(2\sigma)$ for the antithetic estimate we use in experiments. In both cases the perturbation energy is $\mathbb{E}\|\varepsilon\|^2 = D$, which gives the linear dependence on parameter dimension. Antithetic sampling changes only $\Delta$. For PG, the gradient estimate sums $T$ log-probability gradients $\psi_t$, each weighted by the advantage $\hat{A} = R - b$. The variance therefore scales as $\operatorname{Var}(\hat{A})\,\kappa\,T$. Here we match the per-update sample counts of the two methods, $P$ perturbations for ES and $G$ rollouts for PG (Table 1). $G$ is the number of rollouts per update, and when all rollouts share the same input it coincides with the group size of GRPO.

Evaluating Eq. (4) makes the dependence on the action space explicit. The categorical logit score is $u = \mathbf{1}_a - p$, and the Gaussian mean score is $u = (a-\mu)/\sigma_a^2$, which give

$$\text{categorical } (V \text{ symbols}) : \kappa = 1 - \textstyle\sum_v p_v^2 \;\le\; 1 \quad (= 1 - 1/V \text{ for uniform } \pi), \tag{5}$$

$$\text{continuous } (m \text{ dims}) : \kappa = m/\sigma_a^2 \;\propto\; m. \tag{6}$$

Thus the categorical score energy is bounded independently of vocabulary size, while the Gaussian score energy grows with action dimension $m$. Read naively, Eq. (3) says that ES cannot scale to large models. Section 3 shows that only part of each variance affects learning. For ES, this is the part along reward-relevant directions, which scales with the intrinsic dimension rather than $D$. For PG, it is the part that filler tokens add through credit and weight sharing.

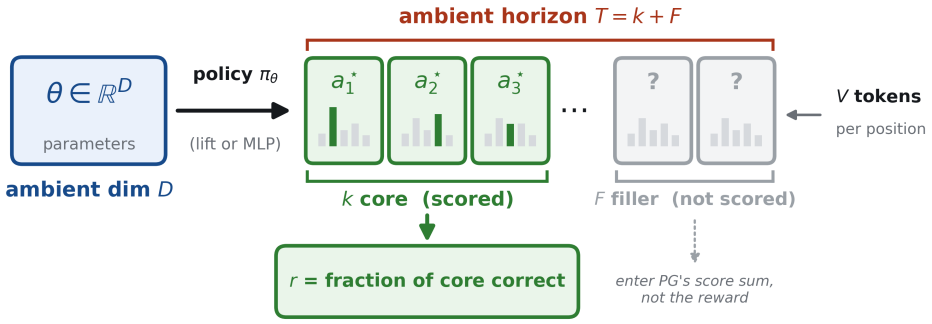


Figure 2: **The controlled testbed** (§2.3). The policy maps parameters $\theta \in \mathbb{R}^D$ to a categorical distribution over $V$ tokens at each of $T = k + F$ positions. The $k$ *core* positions have fixed targets, and the reward is the fraction of them that are correct. The $F$ *filler* positions do not affect the reward but still enter the PG gradient. The four controls are the parameter count $D$, the number of targets $k$, the filler length $F$, and the vocabulary size $V$.

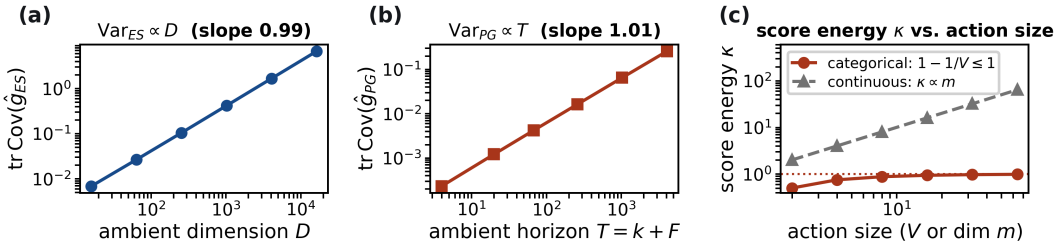


Figure 3: **Variance at a fixed policy** (linear lift at initialization, $k = 4$, $V = 5$). Panels (a) and (b) measure $\mathrm{tr}\,\mathrm{Cov}(\hat{g})$ without optimization. **(a)** ES variance grows linearly with $D$ over a $1000\times$ range. **(b)** PG variance grows linearly with $T$. **(c)** Per-step score energy $\kappa$ is bounded for categorical actions and grows linearly with continuous action dimension.

### 2.3 Controlled testbed and measured scalings

We design a *combination-lock* task (Fig. 2). The policy generates $T = k+F$ tokens from a vocabulary of size $V$. Only the first $k$ tokens affect the reward. Each of them has a fixed target, and the reward is the fraction that are correct. The remaining $F$ filler tokens do not affect the reward, but they are still generated and still enter the PG gradient, analogous to the tokens an LLM generates around a final answer. The reward is given only at the end, as in outcome-reward fine-tuning. This design provides explicit control over the intrinsic dimension through $k$ and over the horizon through $F$, independently of the parameter count $D$ and the vocabulary size $V$. We fix the output length so that the horizon can be set directly. An LLM chooses its own length, so our benchmark probes (§5) use each example's actual length.

We implement the policy in two settings, (1) a *linear lift* and (2) a *nonlinear MLP*. In the linear lift, $\mathrm{logits}_t = A_t\theta$, where each position has its own fixed random matrix $A_t \in \mathbb{R}^{V\times D}$. In the MLP, a single small network maps a position encoding through a `tanh` hidden layer to the $V$ logits, so all positions share the same weights. The two settings serve different purposes. In the linear lift, the reward depends on all $D$ coordinates, yet the intrinsic dimension is exactly $(V-1)k$, so we can vary $D$ and $D_{\mathrm{eff}}$ separately. Because each position has its own readout, the gradient of a filler token overlaps the reward-relevant directions only by chance, and filler affects PG much less than on the MLP. The linear lift therefore isolates the dimension axis. The MLP instead reproduces the weight sharing of an LLM, where the same weights generate every token, and so exposes the horizon cost of PG (§3). It is the setting of our main trained comparison. In every comparison, ES and PG train the same policy, and App. B repeats the trained comparison on the linear lift and on an autoregressive policy.

**Training protocol.** All trained results in the testbed use the same protocol. Each method receives 20k rollouts unless noted otherwise, and we report greedy-decode accuracy on the $k$ reward-carrying tokens. ES uses antithetic, rank-shaped perturbations and scores each perturbed policy on one

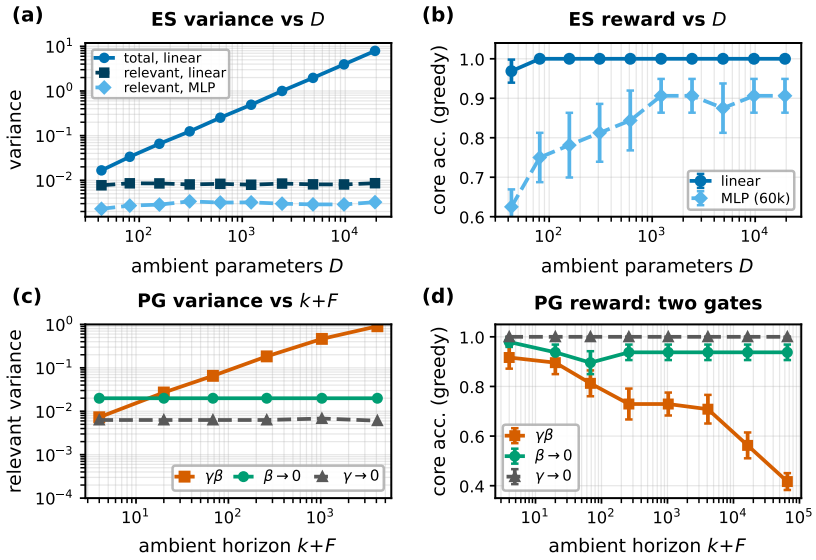


Figure 4: **Dimension is free, horizon is not. (a, b)** ES on both policies, at matched parameter counts. Total gradient variance on the linear lift grows $\propto D$, while the reward-relevant variance is flat on both the linear lift ($D_{\text{eff}} = 16$) and the MLP. Trained accuracy on the linear lift is 1.0 at every $D$. On the MLP it reaches a plateau of about 0.9 at 60k rollouts and falls below it only when a very narrow hidden layer limits capacity. **(c, d)** PG with $k = 4$. Reward-relevant variance grows with the horizon when filler shares both credit and weights ($\gamma\beta$, MLP). It stays flat under dense credit ($\beta \to 0$) or when each position has its own copy of the MLP ($\gamma \to 0$). After 20k matched rollouts, terminal-credit PG with shared weights falls from 0.92 to 0.42.

sampled trajectory, so both methods use the same number of rollouts per update ($P = G = 64$). PG uses REINFORCE with a mean baseline and an entropy bonus that prevents premature convergence to incorrect tokens. We tune the bonus separately for each credit scheme, since per-token credit produces larger updates than a single terminal reward. The bonus is 0.1 for terminal credit and 0.6 for dense credit.

**Measured variance (Fig. 3).** We measure both variances on the linear lift at its initialization $\theta = 0$, where the policy is uniform and $g^\star$ is available in closed form, with $k = 4$ targets and $V = 5$. For ES we vary $D$ from 16 to 16,384 with no filler. For PG, we fix $D = 256$ and vary $F$ from 0 to 4096. The measured covariance traces follow Eq. (3), with fitted slopes of 0.99 for ES against $D$ and 1.01 for PG against $T$. Panel (c) evaluates Eqs. (5)–(6). The categorical term approaches 1, while the continuous term grows linearly with action dimension.

## 3 Effective Dimension and Effective Horizon

We now identify which exploration noise affects reward by projecting each estimate onto the reward-relevant parameter subspace $S$. The effective dimension $D_{\text{eff}}$ counts parameter directions along which reward varies; the effective horizon $T_{\text{eff}}$ counts token positions, weighted by their contribution to reward-relevant PG variance. We relate these counts to Eq. (3), and App. A gives the assumptions and derivation.

**Effective dimension for ES.** Suppose reward depends on the perturbation $\varepsilon$ only through its projection onto a $D_{\text{eff}}$-dimensional subspace. The total and projected variances of the antithetic estimate then scale as

$$\operatorname{tr}\operatorname{Cov}\big(\hat{g}_{\text{ES}}\big) \;\propto\; \mathbb{E}[\Delta^2]\, D, \qquad \underbrace{\operatorname{tr}\operatorname{Cov}\big(P_S\, \hat{g}_{\text{ES}}\big)}_{\text{reward-relevant}} \;\propto\; \mathbb{E}[\Delta^2]\, D_{\text{eff}}, \qquad \mathbb{E}[\Delta^2] \le \tfrac{1}{4\sigma^2}\ (R \in [0,1]). \tag{7}$$

The bound on the perturbation signal $\mathbb{E}[\Delta^2]$ follows from a reward in $[0, 1]$, so increasing $D$ at fixed $D_{\text{eff}}$ adds variance only outside $S$, where it does not change the expected update. In the linear lift, $S$

has dimension exactly $(V-1)k$ by softmax shift invariance, which we confirm numerically, so the number of targets $k$ sets $D_{\text{eff}}$. Figure 4a, b confirms both predictions. The total ES variance grows linearly in $D$ while the projected variance stays flat on both policies, and trained ES accuracy on the linear lift stays flat across a $450\times$ sweep in $D$. The dimension tax is therefore the intrinsic dimension, not the parameter count, and it is small when fine-tuning is intrinsically low-dimensional, echoing the blessing of dimensionality thesis proposed in Liang et al. (2026). "Dimension is free" thus refers to the parameter count at fixed $D_{\text{eff}}$, and a larger $D_{\text{eff}}$ remains costly for ES.

**Effective horizon for PG.** For PG, we apply the same projection to $\hat{g}_{\text{PG}} = \sum_t \hat{A}_t \psi_t$. Under the assumptions in App. A, the variance contributions give

$$\operatorname{tr}\operatorname{Cov}\big(\hat{g}_{\text{PG}}\big) \;\propto\; \operatorname{Var}(\hat{A})\,\kappa\,T, \qquad \underbrace{\operatorname{tr}\operatorname{Cov}\big(P_S\,\hat{g}_{\text{PG}}\big)}_{\text{reward-relevant}} \;\propto\; \operatorname{Var}(\hat{A})\,\kappa\,T_{\text{eff}}, \qquad \operatorname{Var}(\hat{A}) \le \tfrac{1}{4}. \tag{8}$$

The $k$ reward-carrying positions define $S$. A filler position contributes no mean gradient under the reward-independence assumption, but can still contribute variance within $S$. We describe its contribution with two factors, $\gamma$, the fraction of its score energy that lies in $S$, and $\beta$, the relative credit assigned to it. Weight sharing controls the first factor and credit assignment the second, so we refer to them as the sharing gate and the credit gate.

In the uniform case, these factors give $T_{\text{eff}} = k + \gamma\beta F$. In general, each filler position has its own factors, and $T_{\text{eff}} = k + \sum_{\text{filler}} \beta_t \gamma_t \kappa_t$ (App. A). Filler therefore increases the effective horizon only when it both receives credit and contributes score energy along reward-relevant directions, as the table below summarizes.

| | $\gamma \to 0$ (private weights) | $\gamma \approx 1$ (shared weights) |
|---|---|---|
| $\beta \to 0$ (dense credit) | free | free |
| $\beta \approx 1$ (terminal credit) | free | **taxed** |

**Testing the two gates (Fig. 4c, d).** This view predicts that filler degrades PG only when both gates are open. We test this on the MLP by closing one gate at a time. When all positions share one network and the reward arrives only at the end, both gates are open. In this case, PG's reward-relevant variance grows steadily as we add filler, and its trained accuracy drops. Crediting each token separately closes the credit gate, and both the variance and the accuracy stay flat. Giving each position its own copy of the network closes the sharing gate, with the same result. Other common PG estimators such as GRPO and RLOO behave the same way (App. B). ES never assigns credit to individual tokens, so adding filler does not affect it (Fig. 5b).

**Putting the two costs side by side.** Equations (7) and (8) have the same form. Each method's reward-relevant variance is a bounded noise term times a count, which is $D_{\text{eff}}$ for ES and $T_{\text{eff}}$ for PG. Because the noise terms are bounded, how each method's variance scales with task structure is set by these two counts, up to method-specific factors.

$$\text{ES cost} \;\propto\; D_{\text{eff}}, \qquad \text{PG cost} \;\propto\; T_{\text{eff}}. \tag{9}$$

Here, cost means estimator variance. In our testbed, the size of the true gradient stays roughly the same as the horizon grows, and on the linear lift also as $D$ grows (App. A.1). The variance therefore tracks how many samples each method needs. It does not predict absolute performance, which also depends on the training budget and the optimization landscape (§4). On real tasks, directions and tokens matter to different degrees rather than all or nothing, so in §5 we estimate the two counts with probes.

## 4 Trained performance across dimension and horizon

We now ask whether these two costs determine which method performs better after training. We train both methods on the MLP across a grid of effective dimensions and horizons. Throughout this grid, PG receives a single reward at the end of each rollout, so its credit is sparse. Figure 5a shows the difference in final accuracy between ES and PG.

The plane splits into three regions, as the account predicts. When both the dimension and the horizon are small, both methods solve the task and tie. When the horizon is short but the dimension is larger, PG performs better. PG receives a separate learning signal for every token, while ES must find a good direction from a single number per rollout, and this search becomes harder as the dimension

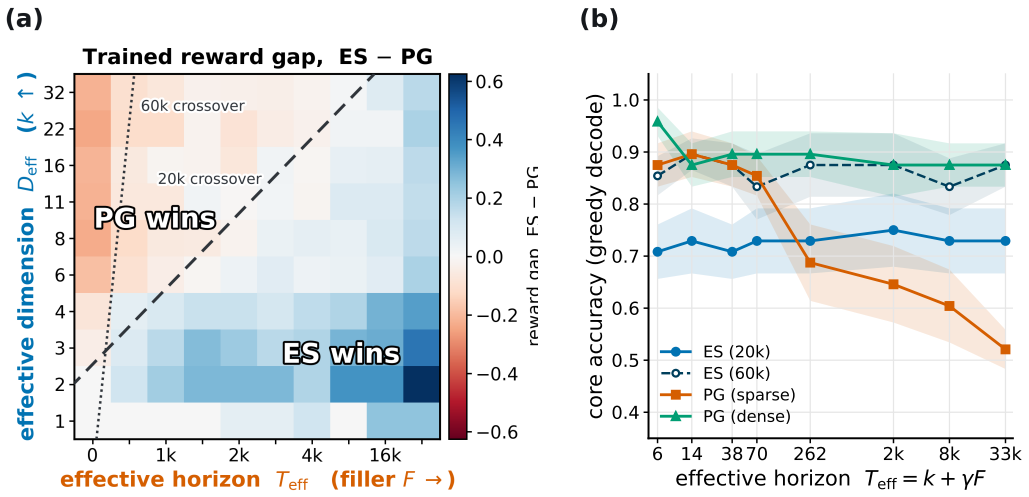


Figure 5: **Trained performance on the MLP policy. (a)** Final accuracy of ES minus PG across dimension and horizon, with PG trained on a single terminal reward (sparse credit). The dashed line marks where the two methods tie at matched budget, and the dotted line where they tie when ES is trained three times longer. **(b)** One row of the plane as the horizon grows, adding PG with dense credit as a control. Sparse-credit PG falls below ES at long horizons, while dense-credit PG stays above it.

grows. As the horizon grows, the filler adds noise to PG's gradient but leaves ES unaffected, and ES overtakes PG. The boundary between the PG and ES regions runs diagonally, because the more directions ES has to search, the longer the horizon must be before its insensitivity to filler becomes an advantage. The boundary also depends on the training budget. ES learns more slowly than PG on this policy and needs about three times as many rollouts to reach its final accuracy. With this larger budget, the region where ES performs better extends to much shorter horizons, so the matched-budget comparison understates the advantage of ES. Without any filler, ES never outperforms PG, even with the larger budget, which the account also predicts, since ES then has no horizon cost to avoid.

Figure 5b follows one row of the plane as the horizon grows and adds dense credit as a control, in which each reward-carrying token is credited separately. With sparse credit, PG starts ahead of ES and falls below it once the horizon is long. With dense credit, PG keeps its short-horizon accuracy at every horizon and stays above ES. Accurate credit assignment therefore removes the horizon cost, which confirms that sparse credit is what gives ES its advantage. An autoregressive policy shows the same pattern, and on the linear lift, where filler affects PG much less, almost every setting ends in a tie (App. B).

## 5 Where real fine-tuning tasks fall

The testbed suggests that ES should gain in LLM fine-tuning when the update a task needs is low-dimensional, which a strong pretrained model makes likely (Aghajanyan et al., 2021; Hu et al., 2022; Liang et al., 2026), and when the reward arrives only at the end of a long generation, as in agentic tasks with outcome rewards. Better credit assignment, such as process rewards (Lightman et al., 2024) or Monte Carlo value estimates (Kazemnejad et al., 2025), should reduce the advantage. To check these predictions, we estimate proxies for the effective dimension and horizon of 27 real tasks with Qwen2.5-7B-Instruct (Qwen Team, 2024) and compare the resulting map with the outcome of training both methods on a subset. The tasks cover language understanding and generation (ARC (Clark et al., 2018), CommonsenseQA (Talmor et al., 2019), HellaSwag (Zellers et al., 2019), BoolQ (Clark et al., 2019), SQuAD (Rajpurkar et al., 2016), WebQuestions (Berant et al., 2013), XSum (Narayan et al., 2018), AG News (Zhang et al., 2015), WinoGrande (Sakaguchi et al., 2020), OPUS-100 translation (Zhang et al., 2020), TriviaQA (Joshi et al., 2017), IMDB (Maas et al., 2011)), mathematical reasoning (GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), MATH-500 (Lightman et al., 2024), Minerva Math (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), AIME 2024, and AMC 2023), the tasks used in prior ES work (Qiu et al., 2026) (Countdown (Gandhi et al., 2024), Sudoku, ARC-AGI (Chollet, 2019), and conciseness), and four agentic environments

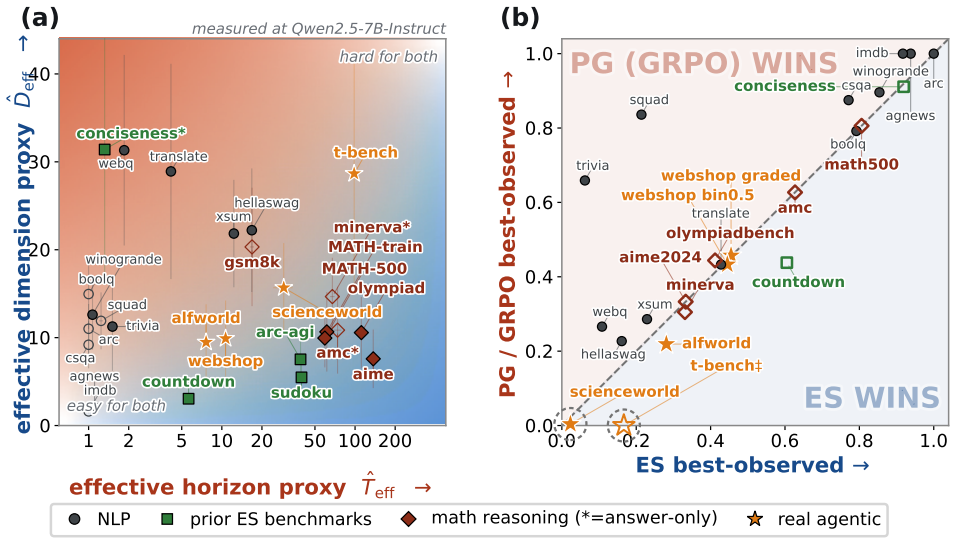


Figure 6: **Benchmark measurements and trained outcomes. (a)** Proxies for $T_{\text{eff}}$ and $D_{\text{eff}}$ on 27 benchmarks at Qwen2.5-7B-Instruct. Vertical bars show one standard deviation across 12 random coordinate subsets, and horizontal bars show one standard error of the mean horizon. Table 3 reports spreads and convergence flags. Hollow markers denote tasks that the base model already solves (at least 81% accuracy), whose dimension estimates are lower bounds. Asterisks mark answer-only targets. Background shading is schematic and is not a fitted decision boundary. **(b)** Best observed performance on 24 of the 27 tasks. Filled markers are our 17 comparisons of full-weight ES with rank-16 LoRA Dr.GRPO (Liu et al., 2025). Appendix C.3 reports the evaluation procedures and controls for the parameterization difference. The 7 hollow markers are read from the figures of prior work (Qiu et al., 2026). They cover Countdown at Qwen2.5-3B, conciseness at 7B, and five math tasks at Qwen2.5-Math-7B. The prior math comparisons use released GRPO, Dr.GRPO, and PPO checkpoints and are not matched training runs. Dashed rings mark ScienceWorld and Terminal-Bench, where neither method reliably improves on the base model.

(ALFWorld (Shridhar et al., 2021), WebShop (Yao et al., 2022), ScienceWorld (Wang et al., 2022), and Terminal-Bench (Merrill et al., 2026)).

**Estimating the effective dimension.** On a real task we estimate $D_{\text{eff}}$ from the directions in which different examples move the model's weights. This estimate is a practical proxy for the reward-relevant dimension of §3, not a direct measurement of it. For each example, we take the reference answer from the dataset, which we call the gold target, and compute the gradient of the model's loss on it. If all examples move the weights in similar directions, the task is low-dimensional, and if they move them in many unrelated directions, it is high-dimensional. The participation ratio of the matrix of inner products between these gradients counts how many directions carry most of the variation. Since it cannot exceed the number of examples, we increase the sample size until the value stabilizes and mark tasks where it does not as lower bounds (App. C.1).

**Estimating the effective horizon.** PG's horizon cost comes from positions where sampling could have produced a different token. For a token distribution $p_t$, the score energy of Eq. (4) is $\kappa_t = 1 - \sum_v p_{t,v}^2$, which is the probability that two independent samples at position $t$ disagree. We compute $\kappa_t$ at every position of the gold target and take the participation ratio of these values, $(\sum_t \kappa_t)^2 / \sum_t \kappa_t^2$. This counts the positions that carry comparable score energy, so $T$ equally uncertain positions give $T$, while a sequence dominated by one uncertain position gives about one. For agentic tasks we run the environment and pool the positions of all turns in an episode. In the testbed this recovers the true horizon. On real tasks it is only a proxy for $T_{\text{eff}}$, because it captures the score energy of each token but not its credit $\beta_t$ or its overlap $\gamma_t$.

**Where the tasks fall.** Figure 6a places the 27 tasks on the plane. Short-answer tasks such as classification and multiple-choice questions have an effective horizon of about one token. Math tasks have the longest horizons, because a model's reasoning chains contain many uncertain steps. The agentic tasks lie in between, since a 7B model chooses most of their actions confidently. Terminal-

Bench, with its long and varied commands, approaches the math tasks. The effective dimension stays small across all 27 tasks, and for math it shrinks further from a 1.5B to a 7B model (App. C.2). The tasks on which ES was previously reported to outperform RL, namely Countdown, Sudoku, and ARC-AGI, are among the lowest-dimensional in the set.

**Head-to-head results.** Figure 6b compares the best accuracy that full-weight ES and PG reached on 17 of the tasks. The short-horizon language tasks favor PG or end in a tie, consistent with their position in the map. ALFWorld, which is low-dimensional, favors ES. On ScienceWorld and Terminal-Bench, neither method improves much on the base model, so these tasks do not test the account. WebShop is a notable mismatch. Its proxy coordinates suggest a regime that favors ES, but training ends in a tie.

These comparisons come with caveats. The seven results from prior work use different models. In particular, its math results use Qwen2.5-Math-7B and compare against released RL checkpoints rather than matched training runs. Our agentic runs are small, with 8 to 20 training iterations evaluated on a few dozen held-out games or goals, 30 ScienceWorld tasks, or 6 Terminal-Bench tasks. They show early training on a small sample, not what either method can reach with a substantial training effort. We also report each run's best checkpoint, which favors ES, since it has no early-stopping signal of its own (App. C.3).

## 6 Related work

We relate our analysis to prior work on exploration and on RL fine-tuning of language models.

**Parameter-space exploration.** ES and related methods are established alternatives to policy-gradient RL (Salimans et al., 2017; Sehnke et al., 2010; Wierstra et al., 2014; Maheswaranathan et al., 2019). Salimans et al. (2017) discuss the advantages of ES on long horizons, and Vemula et al. (2019) analyze parameter dimension, action dimension, and horizon. At LLM scale, Qiu et al. (2026) report competitive ES performance, and Liang et al. (2026) explain its scalability by showing that a small set of high-curvature directions dominates the improvement. Intrinsic-dimension and low-rank adaptation results provide further context (Li et al., 2018; Aghajanyan et al., 2021; Hu et al., 2022). Our work adds a controlled separation of weight sharing and credit assignment, trained comparisons, and measured task coordinates.

**Policy gradients and long-horizon agents.** REINFORCE variants underpin current LLM RL (Williams, 1992; Shao et al., 2024; Guo et al., 2025; Liu et al., 2025). Finer credit assignment through Monte Carlo value estimates can improve sample efficiency (Kazemnejad et al., 2025). Yue et al. (2025) find that several RL methods perform similarly on reasoning and beat the base model only at small $k$ in pass@$k$. Tmax and T1 address instability in terminal-agent training (Ivison et al., 2026; Yang et al., 2026), while TRACE estimates credit for individual tool actions (Tao et al., 2026).

## 7 Conclusion

ES pays for the dimension of the update a task needs, and PG pays for the horizon over which it must assign credit. On a synthetic task that separates the two, ES outperforms PG when the horizon is long relative to the dimension, and accurate credit assignment removes this advantage.

The main implication concerns long-horizon fine-tuning. Agentic tasks with sparse outcome rewards are exactly where PG struggles, and ES incurs no cost from their horizon. Our measurements suggest that the dimension these tasks require stays small. Larger pretrained models are known to need lower-dimensional updates (Aghajanyan et al., 2021). If this trend continues, our account predicts that ES will become increasingly practical on long-horizon tasks. Our head-to-head results are consistent with this picture but too limited to confirm it, and establishing it requires larger training comparisons on long-horizon tasks.

**Limitations.** Our cost account describes estimator variance near a fixed policy. It predicts how costs change within a setting, but not absolute performance, which also depends on the budget and the optimization landscape. The benchmark probes are proxies measured on the base model, and both quantities change during training. The controlled task uses small policies with fixed-length outputs. Our head-to-head comparisons are few, short, and run on small task subsets, and part of the comparison relies on results from prior work with different models. Finally, ES and PG use full weights and LoRA, respectively, in these runs. Restricting ES to the same LoRA subspace does not help it (App. C.3), so a low effective dimension need not mean that the relevant directions lie in a fixed low-rank subspace.

### AI USE STATEMENT

We used generative AI tools extensively as an implementation aid: writing and running the measurement and training code for the controlled testbed, the real-benchmark $D_{\text{eff}}/T_{\text{eff}}$ probes, and the trained ES-vs-PG experiments; orchestrating the GPU experiments; producing and editing figures; and drafting and editing manuscript text throughout, under our direction and iterative review. AI tools were also used, under our direction, to help diagnose and correct a measurement issue in the real-benchmark $D_{\text{eff}}$ probe (App. C); we verified the fix empirically before relying on it, and an earlier, insufficiently verified claim was retracted once a wider check contradicted it. The core scientific contributions – the $D_{\text{eff}}/T_{\text{eff}}$ duality account, the experimental design, and the interpretation of results – were formulated by the authors; AI tools were not used to originate the central hypotheses. We reviewed all AI-assisted code, analysis, text, and figures before inclusion and take responsibility for the final content of this work.

### ETHICS STATEMENT

This work is a controlled synthetic study of two RL gradient estimators plus a measurement of intrinsic-dimension and effective-horizon statistics on public LLM fine-tuning benchmarks. It introduces no new datasets of personal or sensitive data, no human-subjects experiments, and no dual-use capability. We do not foresee direct ethical concerns beyond those already inherent to the cited benchmarks and base models.

### REPRODUCIBILITY STATEMENT

Section 2.3 describes the controlled task, and App. A presents the variance model and its assumptions. Section 4 and Fig. 5 specify the synthetic training comparison and rollout budgets. Appendix B repeats the comparison on the linear lift and on an autoregressive policy. The benchmark probes use gold-target gradients for $D_{\text{eff}}$ and the participation ratio of the score energy for $T_{\text{eff}}$. Section 5 and App. C.1 give their definitions and limitations. Appendix C.2 compares the measured coordinates across the tested Qwen2.5 model sizes, and App. C.3 documents the real-benchmark training comparisons.

## REFERENCES

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7319–7328, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *EMNLP*, pp. 1533–1544, 2013. URL https://aclanthology.org/D13-1160/.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019. URL https://arxiv.org/abs/1911.01547.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019. URL https://arxiv.org/abs/1905.10044.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

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. URL https://arxiv.org/abs/2110.14168.

Kanishk Gandhi, Denise Lee, Gabriel Grand, Muxin Liu, Winson Cheng, Archit Sharma, and Noah D. Goodman. Stream of search (SoS): Learning to search in language. In *COLM*, 2024. URL https://arxiv.org/abs/2404.03683.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, Sept 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *ACL*, 2024. URL https://arxiv.org/abs/2402.14008.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks*, 2021. URL https://arxiv.org/abs/2103.03874.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2106.09685.

Hamish Ivison, Junjie Oscar Yin, Rulin Shao, Teng Xiao, Nathan Lambert, and Hannaneh Hajishirzi. TMax: A simple recipe for terminal agents, 2026. URL https://arxiv.org/abs/2606.23321.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, 2017. URL https://arxiv.org/abs/1705.03551.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 29557–29590, 2025. URL https://proceedings.mlr.press/v267/kazemnejad25a.html.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *NeurIPS*, 2022. URL https://arxiv.org/abs/2206.14858.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations (ICLR)*, 2018. doi: 10.48550/arXiv.1804.08838. URL https://arxiv.org/abs/1804.08838.

Qiyao Liang, Jinyeop Song, Yizhou Liu, Jeff Gore, Ila Fiete, Risto Miikkulainen, and Xin Qiu. The blessing of dimensionality in llm fine-tuning: A variance-curvature perspective, 2026. URL https://arxiv.org/abs/2602.00170.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

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. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=5PAF7PAY2Y.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *ACL*, 2011. URL https://aclanthology.org/P11-1015/.

Niru Maheswaranathan, Luke Metz, George Tucker, Dami Choi, and Jascha Sohl-Dickstein. Guided evolutionary strategies: augmenting random search with surrogate gradients. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4264–4273. PMLR, 2019. URL http://proceedings.mlr.press/v97/maheswaranathan19a.html.

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, et al. Terminal-Bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *ICLR*, 2026. URL https://arxiv.org/abs/2601.11868.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *EMNLP*, 2018. URL https://arxiv.org/abs/1808.08745.

Xin Qiu, Yulu Gan, Conor F. Hayes, Qiyao Liang, Yinggan Xu, Roberto Dailey, Elliot Meyerson, Babak Hodjat, and Risto Miikkulainen. Evolution strategies at scale: LLM fine-tuning beyond reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2026. URL https://arxiv.org/abs/2509.24372.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *EMNLP*, 2016. URL https://arxiv.org/abs/1606.05250.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. In *AAAI*, 2020. URL https://arxiv.org/abs/1907.10641.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning, 2017. URL https://arxiv.org/abs/1703.03864.

Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010. doi: 10.1016/J.NEUNET.2009.12.004. URL https://doi.org/10.1016/j.neunet.2009.12.004.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *ICLR*, 2021. URL https://arxiv.org/abs/2010.03768.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *NAACL*, 2019. URL https://arxiv.org/abs/1811.00937.

Leitian Tao, Baolin Peng, Wenlin Yao, Tao Ge, Hao Cheng, Mike Hang Wang, Jianfeng Gao, and Sharon Li. TRACE: Turn-level reward assignment via credit estimation for long-horizon agents, 2026. URL https://arxiv.org/abs/2607.13988.

Anirudh Vemula, Wen Sun, and J. Bagnell. Contrasting exploration in parameter and action space: A zeroth-order optimization perspective. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 2926–2935. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/vemula19a.html.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In *EMNLP*, 2022. URL https://arxiv.org/abs/2203.07540.

Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *J. Mach. Learn. Res.*, 15(1):949–980, 2014. doi: 10.5555/2627435.2638566. URL https://dl.acm.org/doi/10.5555/2627435.2638566.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Junyao Yang, Yucheng Shi, Zhongzhi Li, Ruhan Wang, Zongxia Li, Haitao Mi, and Leowei Liang. T1: Terminal agent reinforcement learning for long-horizon tasks, 2026. URL https://arxiv.org/abs/2609.11042.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *NeurIPS*, 2022. URL https://arxiv.org/abs/2207.01206.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://arxiv.org/abs/2504.13837.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *ACL*, 2019. URL https://arxiv.org/abs/1905.07830.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. Improving massively multilingual neural machine translation and zero-shot translation. In *ACL*, 2020. URL https://arxiv.org/abs/2004.11867.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NeurIPS*, 2015. URL https://arxiv.org/abs/1509.01626.

# A Derivation of the effective horizon

This appendix derives the effective horizon used in Eq. (8) and states the assumptions it needs. The policy emits $T = k + F$ tokens. The first $k$ core tokens determine the reward, and the $F$ filler tokens do not. The reward $R$ is the fraction of core tokens that match their targets, so it lies in $[0, 1]$. The PG estimate is $\hat{g}_{\text{PG}} = \sum_t \hat{A}_t \psi_t$, where $\psi_t = \nabla_\theta \log \pi(a_t \mid s_t)$ is the score of the token at position $t$ and $\hat{A}_t$ is the advantage assigned to it. The reward-relevant subspace $S$ is spanned by the core scores, and $P_S$ projects onto it.

**Assumptions.** We make three assumptions, and all three hold by construction in the testbed. First, tokens at different positions are sampled independently given the parameters. Second, the reward does not depend on the filler tokens. Third, the advantage given to a filler token does not depend on that token. An autoregressive policy breaks the first assumption, so we test one empirically in App. B.

**Derivation.** Every score has zero mean, because $\mathbb{E}[\psi_t] = \sum_a \nabla_\theta \pi(a) = \nabla_\theta 1 = 0$. Now take any term of the covariance that involves a filler position $t$, either alone or paired with another position $s$. Under the three assumptions, the token $a_t$ is independent of everything else in that term, so its expectation contains the factor $\mathbb{E}[\psi_t] = 0$. Filler therefore contributes no signal and no cross-covariance. It contributes only its own variance, $\mathbb{E}[\hat{A}_t^2]\, \mathbb{E}\|P_S \psi_t\|^2$. The core terms do not depend on $F$, and we normalize them so that each core position counts as one. Dividing each filler term by the contribution of one core position gives

$$\operatorname{tr} \operatorname{Cov}(P_S \hat{g}_{\text{PG}}) \;\propto\; k + \sum_{\text{filler}} \beta_t \gamma_t \kappa_t, \qquad \beta_t = \frac{\mathbb{E}[\hat{A}_t^2]}{\mathbb{E}[\hat{A}_{\text{core}}^2]}, \quad \gamma_t = \frac{\mathbb{E}\|P_S \psi_t\|^2}{\mathbb{E}\|\psi_t\|^2}, \quad \kappa_t = \frac{\mathbb{E}\|\psi_t\|^2}{\mathbb{E}\|\psi_{\text{core}}\|^2}. \tag{10}$$

In the testbed every position reads the parameters through a readout of the same scale. The ratio $\kappa_t$ then equals the ratio of the categorical score energies $1 - \sum_v p_{t,v}^2$ of Eq. (4). When all positions share the same factors, the sum reduces to $T_{\text{eff}} = k + \beta\gamma F$.

**The two gates.** The factor $\beta_t$ is the credit gate. It measures how much advantage a filler token receives relative to a core token. Terminal credit gives every token the same advantage, so $\beta_t = 1$. Per-token credit that gives filler no advantage sets $\beta_t = 0$. The factor $\gamma_t$ is the sharing gate. It is the fraction of the filler token's score energy that falls inside $S$. It is zero when filler and core use disjoint parameters, as in the per-position MLP. When the filler score points in a random direction of the $D$-dimensional parameter space, its expected value is $r/D$ with $r = \dim S$, which is the case of the linear lift. In the shared MLP we measure values between $0.2$ and $0.6$, and they fall as the filler moves away from the core. Both gates lie between $0$ and $1$. Filler therefore lengthens the effective horizon only when it receives credit, shares directions with the core, and is uncertain.

**Scope.** The derivation concerns the variance of one on-policy gradient estimate. It does not cover bias from off-policy updates or trust regions. It also assumes that filler is truly irrelevant to the reward. In a language model, early tokens can change later rewarded tokens, and the autoregressive check in App. B probes this case. For ES, adding positions leaves the variance unchanged only when the added tokens leave the return of every perturbation unchanged.

Table 1 places the two estimators side by side.

## A.1 Signal-to-noise at a fixed policy

How many samples a method needs depends on its noise relative to its signal, not on its noise alone. Table 2 therefore reports, for the conditions of Fig. 4, the reward-relevant signal $\|P_S\, \mathbb{E}\hat{g}\|^2$, the reward-relevant noise $\operatorname{tr} \operatorname{Cov}(P_S \hat{g})$, and their ratio at the initial policy. We estimate the signal from many independent gradient estimates and correct it for their finite number.

For PG, the signal stays flat as the horizon grows. The rising variance therefore lowers the signal-to-noise ratio by the same factor. With dense credit, the noise stays flat and the ratio does not fall. For ES on the linear lift, signal and noise both stay flat as $D$ grows. On the MLP, the noise stays flat with width, but the signal at initialization falls by about a factor of three, even though trained accuracy rises with width. A likely cause is that the perturbation scale is fixed while the initial weight scale

Table 1: **The ES–PG correspondence.** Both reward-relevant variances are a product of reward noise, the energy per unit, and an effective count. The two estimators differ in how the effective count responds to the nominal size. In the testbed, $D_{\text{eff}} = (V-1)k$ does not change with $D$, while $T_{\text{eff}} = k + \gamma\beta F$ grows with $F$ unless one of the gates is closed.

| | ES *(parameter space)* | PG *(action space)* |
|---|---|---|
| explores by | perturbing parameters $\theta \pm \sigma\varepsilon$ | sampling actions $a \sim \pi_\theta$ |
| observes | scalar return $R$ | per-token scores $\psi_t$ |
| samples per update | $P$ perturbations | $G$ rollouts |
| energy per unit | $1/\sigma^2$ (tunable) | $\kappa = 1 - \sum_v p_v^2 \le 1$ |
| reward noise | $\mathrm{Var}(R) \le \frac{1}{4}$ | $\mathrm{Var}(\hat{A}) \le \frac{1}{4}$ |
| nominal count | $D$ parameters | $T$ positions |
| effective count | $D_{\text{eff}}$ | $T_{\text{eff}}$ |
| growth with nominal count | none | $\gamma\beta$ per added position |

shrinks with width, but we have not tested this. Quantities measured at the initial policy, including the benchmark probes of §5, describe where training starts. On a nonlinear policy they need not predict how performance changes during training.

Table 2: Reward-relevant signal, noise, and signal-to-noise ratio at the initial policy (means over 3 seeds), at the smallest and largest size of each sweep.

| | | | | SNR | |
|---|---|---|---|---|---|
| condition | sweep | signal | noise | smallest | largest |
| ES, linear lift | $D = 16 \to 16\text{k}$ | flat | flat | 0.90 | 0.86 |
| ES, MLP | $D = 157 \to 9.7\text{k}$ | 2.7× lower | flat | 2.56 | 0.87 |
| PG, MLP, terminal | $F = 0 \to 1024$ | flat | 63× higher | 20.3 | 0.29 |
| PG, linear lift | $F = 0 \to 1024$ | flat | 6.7× higher | 16.7 | 2.5 |
| PG, MLP, dense | $F = 0 \to 1024$ | flat | flat | 111 | 111 |

## B Robustness of the synthetic comparison

We repeat the trained comparison of Fig. 5 on two other policies and compare several PG estimators.

**Linear lift.** Figure 7 repeats the grid of Fig. 5a with the linear lift. Each position reads the shared parameters through its own random readout, so filler scores overlap $S$ only by chance. Filler therefore adds much less reward-relevant variance than on the MLP, and the two methods tie almost everywhere. ES leads only at very long horizons.

**Autoregressive policy.** In Fig. 8 the policy conditions on the tokens it has already generated, as a language model does. The filler comes first and the reward-carrying tokens last. The results keep the same shape. Terminal-credit PG still falls below ES as the horizon grows, although the crossover arrives later.

**Other policy-gradient estimators.** We compare REINFORCE, RLOO (Ahmadian et al., 2024), GRPO (Shao et al., 2024), and Dr.GRPO (Liu et al., 2025), each with one advantage per trajectory. Their total variances differ by up to a factor of 100, but the share of that variance lying in the reward-relevant subspace agrees within 5%. Swapping RLOO for GRPO on the linear-lift grid leaves the preferred method unchanged in every cell. The main experiments use REINFORCE with a mean baseline.

## C The benchmark probes and training comparisons

This appendix defines the probes behind Fig. 6a, reports how their values change with model size, and describes the training comparisons of Fig. 6b.

### C.1 Probe definitions

Both probes use the base model before fine-tuning. They need a forward pass per example, and the dimension probe also needs a backward pass. They use gold targets, meaning the reference answer, reasoning, or expert action provided with the benchmark. They need no reward model and no samples from the policy.

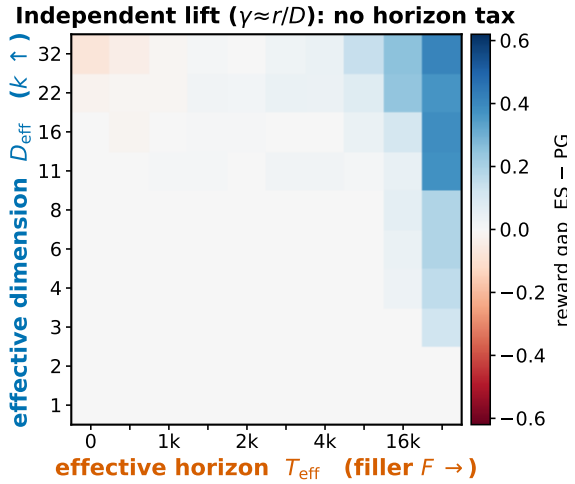


Figure 7: **Linear lift.** Same grid, training protocol, and color scale as Fig. 5a (8 seeds). ES reaches its best accuracy within a few hundred rollouts and nearly all cells are ties. ES leads only beyond about 16k filler positions.

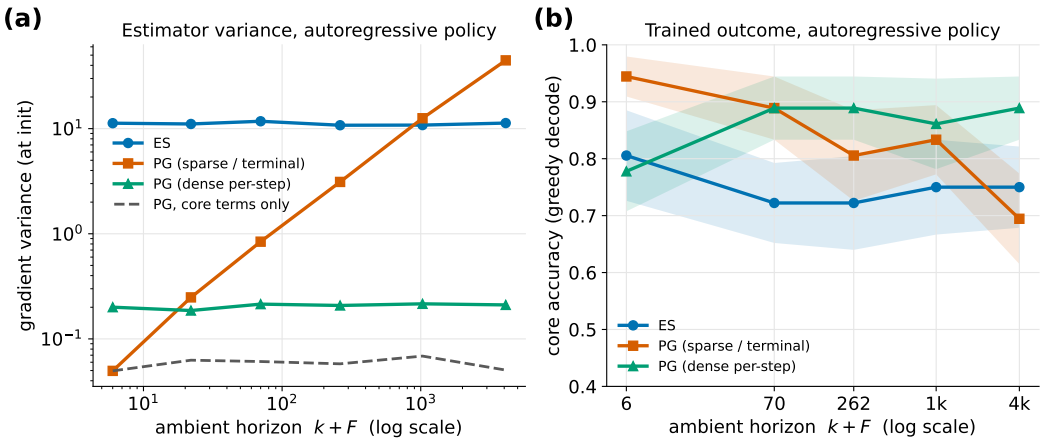


Figure 8: **Autoregressive policy.** The shared network conditions on position, the previous token, and a summary of the sampled prefix. Filler is generated first and the $k$=6 reward-carrying tokens last. **(a)** Gradient variance at initialization. ES and dense-credit PG stay flat, while terminal-credit PG grows with the horizon. The core-token terms of PG stay flat on their own, so all of the growth comes from filler. **(b)** Trained accuracy under the protocol of Fig. 5. Terminal-credit PG falls below ES about ten times later in the horizon than with the positional policy.

**Per-example gradient.** For example $i$ with prompt $x_i$ and gold target $y_i^\star$, we take the average token cross-entropy and its gradient at the base weights $\theta_0$:

$$\ell_i(\theta) = -\frac{1}{|y_i^\star|}\sum_t \log \pi_\theta\big(y_{i,t}^\star \mid x_i, y_{i,<t}^\star\big), \qquad g_i = \nabla_\theta\, \ell_i(\theta_0). \tag{11}$$

This gradient describes how the task moves the weights. It is not the gradient of a particular RL estimator, whose tokens would be sampled from the model rather than taken from the reference.

**Effective dimension.** The full gradient has billions of entries, so we keep a fixed random subset of 8000 coordinates, the same for every example of a task. From the matrix of inner products between these reduced gradients, $K_{ij} = \langle \tilde{g}_i, \tilde{g}_j \rangle$, we take the participation ratio of its eigenvalues,

$$D_{\text{eff}} = \frac{\big(\sum_k \lambda_k\big)^2}{\sum_k \lambda_k^2} = \frac{(\operatorname{tr} K)^2}{\|K\|_F^2}. \tag{12}$$

The ratio is 1 when all gradients are parallel and $n$ when $n$ gradients are orthogonal with equal norms. Because it cannot exceed $n$, we use $n = 256$ examples where the dataset allows and check that the value has leveled off. Tasks whose value is still rising at $n = 256$ are marked as lower bounds. This happens mostly on short-answer tasks that the 7B model already solves, whose gradients are small and example-specific. We repeat the measurement with 12 independent random coordinate subsets. The reported value is their mean, and the error bar is their spread.

**Effective horizon.** From the same forward pass, let $p_t$ be the next-token distribution at position $t$ of the gold target. We compute

$$\kappa_t = 1 - \sum_{v=1}^{V} p_{t,v}^2, \qquad \tau_i = \frac{\big(\sum_t \kappa_t\big)^2}{\sum_t \kappa_t^2}, \qquad T_{\text{eff}} = \frac{1}{n}\sum_i \tau_i. \tag{13}$$

Here $\kappa_t$ is the score energy of Eq. (4). The ratio $\tau_i$ is 1 when one position carries all the score energy and $T$ when $T$ positions carry equal amounts. It ignores the overall size of the energy, which enters the variance separately in Eq. (8). When the model is certain of every token of a target, every $\kappa_t$ is zero and the ratio is undefined. This happens on one-token answers that the 7B model already knows, and we then set $\tau_i = 1$, since the answer still occupies one position.

**Error bars.** Table 3 lists both coordinates with their spreads. The spread of the dimension probe comes from the 12 coordinate subsets, and the spread of the horizon probe comes from the examples. Neither captures systematic error from using gold targets or from the factors the probes leave out. We therefore use the atlas to compare the positions of tasks, and we do not fit a decision boundary to it. Small competition-math sets, such as `minerva` and `amc`, keep wide error bars even with larger coordinate subsets, because they have few examples.

**Agentic episodes.** For an agentic task the horizon probe treats a whole episode as one sequence. It pools the score energies of the gold actions from every turn before taking the participation ratio, since credit is spread across the whole episode. The dimension probe instead uses one gradient per turn, so that each turn counts as one example, as a single prompt does for a static task.

### C.2 The atlas across model size

Table 3 gives both probes at 1.5B and 7B, and Fig. 9 shows how each task moves between the two sizes. The math tasks become markedly lower-dimensional at 7B, while the agentic tasks rise but stay below 30. The measured dimension is therefore small compared with the parameter count at both sizes, but it depends on the model and does not always shrink with scale. The measured horizon shrinks or stays at its floor on nearly every task, because a larger model is more certain of the reference tokens.

### C.3 Training comparisons

Figure 6b compares the two methods after training on each task. This section describes the setup and the evaluation, a control that puts both methods in the same subspace, and the agentic runs.

**Setup.** ES perturbs all of the roughly 7.6B weights of Qwen2.5-7B-Instruct and scores each perturbation with one greedy completion. PG trains rank-16 LoRA adapters on the attention projections with Dr.GRPO, sampling 12 completions per prompt. With a population of 12, both methods use the same number of completions per prompt and update. For every run we report the best evaluation score over its checkpoints. This favors ES, which has no early-stopping signal of its own.

Table 3: Atlas coordinates at Qwen2.5-1.5B and 7B (mean ± sd). The spread of $D_{\text{eff}}$ is over 12 coordinate subsets and the spread of $T_{\text{eff}}$ over examples or episodes. Fig. 6 plots the 7B columns. °The 7B base model already scores at least 81%, and $D_{\text{eff}}$ is a lower bound. $^{\uparrow}D_{\text{eff}}$ is still rising at $n = 256$.

| task | group | $T_{\text{eff}}$ 1.5B | $D_{\text{eff}}$ 1.5B | $T_{\text{eff}}$ 7B | $D_{\text{eff}}$ 7B |
|---|---|---|---|---|---|
| terminalbench | agentic | $169.3 \pm 112.8$ | $23.5 \pm 6.0$ | $98.5 \pm 39.8$ | $28.7 \pm 13.6$ |
| scienceworld | agentic | $47.5 \pm 31.1$ | $10.0 \pm 2.1$ | $29.1 \pm 20.5$ | $15.7 \pm 5.1$ |
| webshop | agentic | $18.5 \pm 8.7$ | $3.5 \pm 0.8$ | $10.7 \pm 5.5$ | $9.9 \pm 4.3$ |
| alfworld | agentic | $22.7 \pm 14.4$ | $3.7 \pm 0.9$ | $7.6 \pm 6.0$ | $9.5 \pm 4.3$ |
| aime | math | $160.2 \pm 39.7$ | $13.6 \pm 3.7$ | $136.9 \pm 38.1$ | $7.6 \pm 3.3^{\uparrow}$ |
| olympiad | math | $136.0 \pm 57.6$ | $28.8 \pm 13.4$ | $111.5 \pm 48.9$ | $10.6 \pm 4.1$ |
| amc | math | $97.9 \pm 31.4$ | $14.2 \pm 2.7$ | $73.9 \pm 25.9$ | $10.8 \pm 4.9^{\uparrow}$ |
| minerva | math | $84.5 \pm 28.1$ | $20.4 \pm 6.9$ | $67.6 \pm 21.9$ | $14.7 \pm 4.4^{\uparrow}$ |
| math | math | $79.7 \pm 48.7$ | $22.3 \pm 11.7$ | $61.4 \pm 38.4$ | $10.7 \pm 4.5$ |
| math500 | math | $79.0 \pm 46.6$ | $22.8 \pm 9.8$ | $59.3 \pm 36.8$ | $9.9 \pm 3.3$ |
| gsm8k | math | $33.9 \pm 17.4$ | $34.3 \pm 16.5$ | $16.9 \pm 9.2$ | $20.3 \pm 6.7^{\circ}$ |
| sudoku | prior ES | $71.3 \pm 10.9$ | $7.9 \pm 2.5$ | $39.6 \pm 11.2$ | $5.5 \pm 1.8$ |
| arcagi | prior ES | $48.5 \pm 41.3$ | $7.4 \pm 1.5$ | $39.0 \pm 37.4$ | $7.5 \pm 2.6^{\uparrow}$ |
| countdown | prior ES | $12.0 \pm 2.6$ | $6.4 \pm 2.1$ | $5.6 \pm 1.8$ | $3.0 \pm 0.5$ |
| conciseness | prior ES | $1.5 \pm 0.6$ | $18.7 \pm 10.7$ | $1.3 \pm 0.6$ | $31.4 \pm 14.3^{\uparrow}$ |
| hellaswag | NLP | $21.2 \pm 7.5$ | $9.7 \pm 3.4$ | $16.8 \pm 6.6$ | $22.2 \pm 7.0$ |
| xsum | NLP | $18.4 \pm 4.2$ | $4.5 \pm 1.4$ | $12.3 \pm 3.7$ | $21.8 \pm 6.1$ |
| translate | NLP | $7.5 \pm 5.8$ | $11.8 \pm 6.0$ | $4.2 \pm 3.9$ | $28.9 \pm 12.2^{\uparrow}$ |
| webq | NLP | $2.6 \pm 2.0$ | $2.6 \pm 0.6$ | $1.9 \pm 1.4$ | $31.3 \pm 10.8^{\uparrow}$ |
| trivia | NLP | $2.0 \pm 1.1$ | $18.6 \pm 8.4$ | $1.5 \pm 0.7$ | $11.3 \pm 5.0$ |
| squad | NLP | $1.5 \pm 0.9$ | $10.1 \pm 3.7$ | $1.2 \pm 0.7$ | $11.9 \pm 3.2^{\circ}$ |
| agnews | NLP | $1.2 \pm 0.4$ | $3.8 \pm 0.7$ | $1.1 \pm 0.2$ | $2.0 \pm 0.2^{\circ}$ |
| winogrande | NLP | $1.1 \pm 0.2$ | $14.0 \pm 5.8$ | $1.1 \pm 0.2$ | $12.6 \pm 4.0$ |
| boolq | NLP | $1.0 \pm 0.0$ | $9.3 \pm 2.6$ | $1.0 \pm 0.0$ | $15.0 \pm 4.0^{\circ\uparrow}$ |
| imdb | NLP | $1.0 \pm 0.0$ | $4.0 \pm 1.0$ | $1.0 \pm 0.0$ | $1.6 \pm 0.2^{\circ}$ |
| csqa | NLP | $1.0 \pm 0.0$ | $17.5 \pm 5.0$ | $1.0 \pm 0.0$ | $9.2 \pm 2.7^{\circ}$ |
| arc | NLP | $1.0 \pm 0.0$ | $18.2 \pm 5.4$ | $1.0 \pm 0.0$ | $11.0 \pm 2.8^{\circ\uparrow}$ |

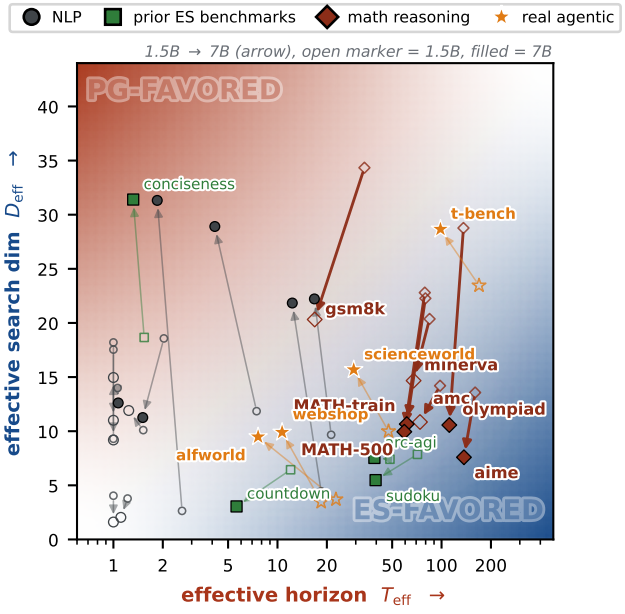


Figure 9: **Benchmark coordinates at 1.5B and 7B.** Arrows run from each task's 1.5B position (open) to its 7B position (filled). The horizon shrinks or stays at its floor. The math tasks lose most of their dimension, while the agentic tasks gain some. Table 3 lists the values.

**Evaluation.** Our first evaluation harness truncated long prompts, left out the answer choices of multiple-choice tasks, cut off generation too early, and compared answers without normalizing case or punctuation. Each of these penalized verbose answers and so favored whichever method learned the answer format first. We corrected all four and report only corrected scores. Table 4 gives them for the six NLP tasks. PG wins five, and `boolq` is a tie at the base score. The probes keep the original prompts.

Table 4: Six NLP comparisons at Qwen2.5-7B-Instruct with the corrected evaluation. Chance is uniform guessing where there is a finite label set.

| task | chance | base | PG | ES | winner |
|---|---|---|---|---|---|
| `agnews` | 0.25 | 0.833 | 1.000 | 0.938 | PG |
| `winogrande` | 0.50 | 0.750 | 0.896 | 0.854 | PG |
| `boolq` | 0.50 | 0.792 | 0.792 | 0.792 | tie |
| `squad` | – | 0.199 | 0.836 | 0.214 | PG |
| `hellaswag` | – | 0.150 | 0.227 | 0.161 | PG |
| `xsum` | – | 0.215 | 0.286 | 0.229 | PG |

**Matched subspace.** Because ES perturbs all weights while PG trains LoRA adapters, we also run ES inside the same LoRA subspace, which is about 750 times smaller. We tuned the perturbation scale separately for this subspace. Table 5 shows that the smaller space does not help ES. On four tasks PG improves while ES stays at its base score. The perturbations do produce different rewards, so the failure does not come from a flat reward signal. We set aside `agnews`, where the training reward saturates, and `boolq`, where neither method improves. This control scores answers by token $F_1$, so its numbers are not directly comparable with Table 4. We therefore compare each method with its own base score. Reducing ES to this low-rank subspace does not improve it. One reading is that a task can need only a few directions without those directions aligning with a fixed LoRA parameterization, so a low effective dimension need not favor a pre-specified low-rank subspace.

Table 5: Both methods in the same rank-16 LoRA subspace of Qwen2.5-7B-Instruct, with the same reward and budget. Gains are relative to the base score in this evaluation.

| task | base | PG peak | ES peak | PG gain | ES gain | note |
|---|---|---|---|---|---|---|
| `squad` | 0.270 | $0.810 \pm 0.040$ | $0.269 \pm 0.000$ | $+0.540$ | $-0.001$ | – |
| `winogrande` | 0.750 | $0.833 \pm 0.040$ | $0.750 \pm 0.000$ | $+0.083$ | $+0.000$ | – |
| `xsum` | 0.211 | $0.275 \pm 0.000$ | $0.211 \pm 0.000$ | $+0.064$ | $+0.000$ | – |
| `hellaswag` | 0.131 | $0.183 \pm 0.010$ | $0.129 \pm 0.010$ | $+0.052$ | $-0.002$ | – |
| `agnews` | 0.833 | $1.000 \pm 0.000$ | $0.854 \pm 0.020$ | $+0.167$ | $+0.021$ | train fitness saturated |
| `boolq` | 0.792 | $0.729 \pm 0.100$ | $0.792 \pm 0.000$ | $-0.063$ | $+0.000$ | neither moves |

**Agentic tasks.** On ALFWorld ES reaches 0.28 and PG 0.22. The PG value is the best of four learning rates, while ES uses one setting, so the tuning effort is not matched. WebShop does not show the ES advantage its position in the atlas suggests. Its partial-credit reward and short successful episodes may make credit assignment easier than the probe implies, but we have not tested this. ScienceWorld and Terminal-Bench are inconclusive. On ScienceWorld, PG does not improve on the base score because every rollout in a group receives the same reward, and ES improves only slightly and not on every seed. On Terminal-Bench, ES solves one of six tasks and PG none. These cases show a limit of the variance account. A method first has to see some reward signal before its variance matters. Both are marked separately in Fig. 6b.