# ESSA: Evolutionary Strategies for Scalable Alignment

**Anonymous Authors**[1]

## Abstract

Alignment of Large Language Models (LLMs) typically relies on Reinforcement Learning from Human Feedback (RLHF) with gradient-based optimizers such as Proximal Policy Optimization (PPO) or Group Relative Policy Optimization (GRPO). While effective, these methods require complex distributed training and careful hyperparameter tuning, which becomes increasingly difficult at scale. We present ESSA, Evolutionary Strategies for Scalable Alignment, a gradient-free post-SFT alignment stage that replaces the online RL optimization loop with inference-only black-box optimization. ESSA optimizes only the singular values of low-rank adapters, making evolutionary search practical even for very large models and compatible with INT4/INT8 inference. Across benchmarks, ESSA matches GRPO and in our best configurations improves test accuracy by up to 21.9 percentage points (e.g., on MATH500, Qwen2.5-7B), while using only forward passes during the alignment stage. At scale, ESSA shows stronger scaling than gradient-based methods: Qwen2.5-32B trained on PRM800K reaches a fixed accuracy threshold on MATH500 $2\times$ faster on 16 GPUs and $6\times$ faster on 128 GPUs than GRPO. These results position evolutionary strategies as a compelling, hardware-friendly alternative to gradient-based LLM alignment, combining competitive quality with substantially reduced wall-clock time and engineering overhead.

## 1. Introduction

Large Language Models (LLMs) have made significant progress thanks to post-training techniques that guide the model's behavior under an explicit downstream criterion. Online RLHF methods such as GRPO, PPO, and RLOO remain the de facto standard (Ouyang et al., 2022b; Schulman et al., 2017; Ahmadian et al., 2024; Shao et al., 2024b), but require actor-critic training, long rollouts, and heavy distributed synchronization. As models grow larger, practical considerations about how to distribute components across GPUs become critical, with sensitivity to numerous interacting hyperparameters and communication bottlenecks (Zheng et al., 2023; Sheng et al., 2024).

To address these issues, we revisit evolutionary strategies as a scalable, gradient-free alternative requiring only forward passes and scalar reward aggregation. We introduce **ESSA (Evolutionary Strategies for Scalable Alignment)**, which pairs Evolutionary Strategies (ES) with parameter-efficient adaptation. We limit optimization to low-rank adapters of attention matrices (Q/K/V/O) and further compress them via singular value decomposition (SVD) parameterization of singular values, making black-box search practical and interpretable (Hu et al., 2021; Vaswani et al., 2017).

In this work, ESSA is introduced as a gradient-free online alignment procedure applied strictly after a supervised fine-tuning (SFT) warm-start. The method replaces only the post-SFT stage. Because ESSA's loop is inference-only, it can evaluate candidates with weight-only INT8/INT4 quantization of the frozen backbone, while keeping the small LoRA modules in BF16. In our setup this enables one-candidate-per-GPU evaluation for models up to 72B parameters on an 80GB GPU. Across mathematics of varying difficulty, instruction following, and general assistant tasks, ESSA matches baselines trained by GRPO, while offering stronger system scalability and reduced hyperparameter fragility (Shao et al., 2024b). Overall, ESSA turns alignment into a simple, highly parallel inference-only optimization loop.

## 2. Related Works

**Alignment.** Alignment of large language models is commonly based on RLHF (Ouyang et al., 2022b), typically optimized with PPO (Schulman et al., 2017) or REINFORCE (Sutton et al., 1999). Variants such as RLOO (Ahmadian et al., 2024), GRPO (Shao et al., 2024b), REINFORCE++ (Hu, 2025) and DAPO (Yu et al., 2025) stabilize training but retain high memory cost. Offline preference learning (Rafailov et al., 2024; Hong et al., 2024; Meng et al., 2024)

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

removes online rollouts but is bounded by dataset coverage and preference noise, limiting generalization (Tang et al., 2024; Xu et al., 2024a).

**Parameter-efficient training of LLMs.** Parameter-efficient fine-tuning (PEFT) techniques reduce the cost of adapting large language models by updating only a small subset of parameters. Beyond classic approaches such as adapters (Houlsby et al., 2019), prefix-tuning (Li & Liang, 2021), and LoRA (Hu et al., 2021), more recent methods include DoRA (Liu et al., 2024), VeRA (Kopiczko et al., 2024), and tensor-based approaches like LoTR (Bershatsky et al., 2024). Another approach optimizes only singular values is Transformer$^2$ (Sun et al., 2025), which also inspired our method. However, gradient-based optimization of singular values is limited, as demonstrated by this method's results on mathematical, coding, and reasoning tasks, and as analyzed by us in 4.4.5.

**Evolution strategies.** ES, including Covariance Matrix Adaptation (CMA-ES) (Hansen & Ostermeier, 2001), Natural Evolution Strategies (NES) (Wierstra et al., 2011), Augmented Random Search (ARS) (Mania et al., 2018), and Guided Evolutionary Strategies (GES) (Maheswaranathan et al., 2018), provide powerful gradient-free optimization that is highly parallel and robust to sparse rewards (Salimans et al., 2017). Zero-order optimizers (Zhang et al., 2024) approximate gradients of loss differences but tend to lag in complex reasoning tasks. Applications of ES to large language model alignment remain rare due to high-dimensionality challenges: existing works like GENOME/GENOME+ (Zhang et al., 2025), LoRAHub (Huang et al., 2024), and DFO (Jin et al., 2024) reduce parameter space but are still limited to a variety of experiments.

**ES for LLM.** ES has recently been applied to LLM fine-tuning (ES-FT (Qiu et al., 2025), EGGROLL (Sarkar et al., 2025)), but existing methods either optimize all parameters or focus on hyperscale regimes. In contrast, ESSA restricts optimization to SVD-parameterized LoRA singular values and targets post-SFT alignment under practical rollout budgets. As summarized in Table 1, ESSA induces the strongest reduction in search dimension among ES-based methods, which is known to be a key factor for improving the stability and sample efficiency of black-box optimization (Salimans et al., 2017). We additionally compare ESSA with ES-FT and EGGROLL on Countdown (Appendix G), where ESSA matches ES-FT while using a much smaller search space.

In this work, however, we focus on real-world LLM alignment benchmarks that are widely adopted by the community (e.g., GSM8K, PRM800K, IFEval, HelpSteer2), and therefore do not include ES-FT and EGGROLL in the main experimental section, as their primary evaluation settings and computational regimes differ substantially from the post-training alignment scenario considered here.

*Table 1.* Comparison of ES-based methods for LLM fine-tuning. ESSA operates in a drastically smaller search space and targets post-SFT alignment, whereas ES-FT and EGGROLL focus on full-parameter or hyperscale ES in different computational regimes.

| Method | Search dimension | Base architecture | Tasks reported |
|---|---|---|---|
| ESSA (ours) | $\mathcal{O}(r)$ | transformer | GSM8K, PRM800K, IFEval, HelpSteer2, Countdown |
| ES-FT | $\mathcal{O}(nm)$ | transformer | Countdown |
| EGGROLL | $\mathcal{O}(r(n+m))$ | recurrent model | GSM8K, Countdown |

# 3. Evolutionary Strategies for Scalable Alignment

## 3.1. Motivation

Modern online alignment of LLMs is dominated by gradient-based RLHF variants. In practice, these pipelines are costly: they require long rollouts, backpropagation through large contexts, optimizer-state synchronization across devices, and careful hyperparameter tuning. As model size grows, memory pressure and training fragility increase, and sparse or noisy rewards further destabilize learning.

ESSA offers the opposite trade-off: it restricts learning to low-rank LoRA adapters and uses a compact SVD-based parameterization in which only a small number of per-direction scaling coefficients are optimized. In this setting, evolutionary updates require nothing beyond forward evaluations under parameter perturbations together with aggregated scalar rewards, which yields a naturally parallelizable, memory-efficient training loop. The standard drawback of evolutionary methods, namely poor efficiency in extremely high-dimensional search spaces, is addressed here by forcing the search to remain within a compact and task-aligned low-rank subspace.

Crucially, ESSA is not a full training pipeline: it is a gradient-free alignment stage applied on top of a SFT initialization. The SFT warm-start preserves model expressivity and places ES in a task-aware region of parameter space. Without SFT, the LoRA matrices begin as an unstructured Gaussian matrix paired with a zero matrix, whose singular vectors are effectively identity-based and uninformed. We analyze the role of SFT initialization in Appendix E (MNIST (LeCun & Cortes, 2005) toy example). SFT is also inexpensive compared to online RL alignment. Prior work reports that SFT requires only a small fraction of the compute used for RL-based alignment (Ouyang et al., 2022a). The expensive component of alignment is the online RL loop, and ESSA eliminates this phase while preserving the lightweight warm-start. Experiments without SFT are provided in Appendix F.1 and demonstrate very important result, that ESSA can operate without SFT, while being

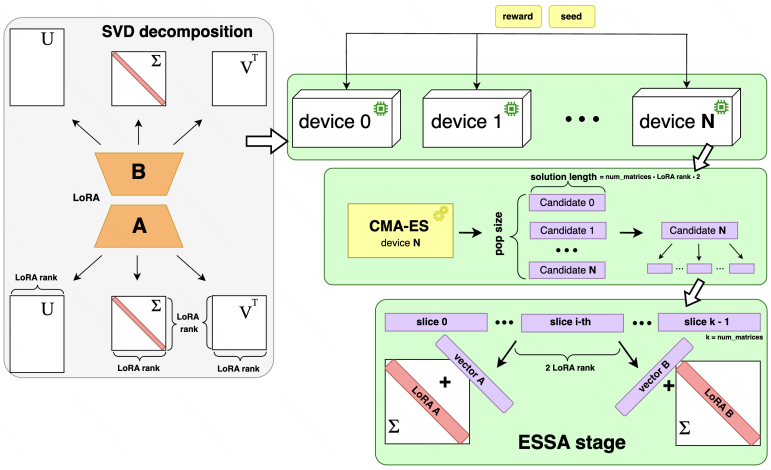



*Figure 1.* Illustration of the ESSA framework. LoRA adapters are first initialized via SFT and decomposed into fixed SVD bases with trainable singular values. The term device N denotes the GPU worker in distributed evaluation. CMA-ES receives a seed at each device, generates a population of size N+1 locally, evaluates a different candidate, and returns a reward. Each candidate is a vector $\Sigma_i \in \mathbb{R}^{\textbf{solution_length}}, i = 0, 1, \ldots,$ N and is added to SVD vectors of the training matrices. It is partitioned into contiguous slices, each of which corresponds to one LoRA matrix (e.g. $W_Q, W_K, W_V, W_O$ for each transformer layer) and contains $2 \cdot$ **LoRA_rank** singular values (for factors $A$ and $B$). The solution length is the dimensionality of the candidate vector, i.e., the concatenation of all perturbation of the trainable LoRA singular values across all matrices and layers. With the number of layers (**num_layers**), number of matrices per layer (**num_matrices_per_layer**), LoRA_rank: **num_matrices** $=$ **num_layers** $\cdot$ **num_matrices_per_layer** and **solution_length** $=$ **num_matrices** $\cdot$ **LoRA_rank** $\cdot$ 2.

primarily intended as a gradient-free replacement for the online alignment phase.

### 3.2. Method

**Initialization.** First, we reduce optimization problem to low-rank updates of each attention matrix $W_0 \in \mathbb{R}^{m\times n}$:

$$\Delta W = BA, \qquad B \in \mathbb{R}^{m\times r},\ A \in \mathbb{R}^{r\times n},\ r \ll \min(m, n).$$

Then, we run a short SFT stage to initialize the LoRA adapters with task-aware parameters. For every backbone-task pair in the paper, we train exactly one SFT LoRA adapter.

**SVD.** To shrink the trainable space even further, we decompose each SFT-LoRA factor separately:

$$A = U_A \Sigma_A V_A^\top, \qquad B = U_B \Sigma_B V_B^\top.$$

The orthogonal matrices $U_A, V_A, U_B, V_B$ are kept fixed after the initial SFT step, while only the top singular values in $\Sigma_A$ and $\Sigma_B$ remain trainable. This parameterization freezes the SFT-learned directions and optimizes only their per-direction scaling, reducing the search dimension from $\mathcal{O}(r(m+n))$ to $\mathcal{O}(r)$ per factor.

**Evolutionary Optimization.** We use CMA-ES (Hansen & Ostermeier, 2001) as the optimizer, maintaining and updating a multivariate normal search distribution over the selected singular values. Each ESSA iteration proceeds as follows (more detailed algorithm is provided in Figure 1 and in Appendix A):

1. CMA-ES samples $\lambda \geq 2$ (population size) candidate singular-value vectors.

$$x_{i+1}^{(k)} \sim m_i + \sigma_i \mathcal{N}(0, C_i)$$

for $\quad k = 1, ..., \lambda, \quad$ so that $\quad x_{i+1}^{(k)} \sim \mathcal{N}(m_i, \sigma_i^2 C_i),$

where $m_i \in \mathbb{R}^n$ is a current mean of the search distribution; $\sigma_i \in \mathbb{R} > 0$ is an "overall" standard deviation; $C_i \in \mathbb{R}^{n\times n}$ is a covariance matrix encoding anisotropic search directions; $n = \text{solution_length}$. Up to the scalar factor $\sigma_i^2$, $C_i$ is the covariance matrix of the search distribution

2. For each candidate, we reconstruct $A$ and $B$ by adding the candidate's singular-value offsets to the fixed SVD decomposition, forming updated low-rank factors and computing $\Delta W = BA$, and evaluate the model on the alignment objective to obtain a scalar reward.

3. After all candidates are evaluated in parallel, CMA-ES updates $m_i$, $\sigma_i$, and $C_i$. This allows the search distribution to gradually align itself with beneficial directions in the objective landscape. Communication between workers is limited to random seeds and scalar rewards, allowing near-linear scaling across many GPUs.

A formal analysis of ESSA computational and communication properties and time efficiency is provided in Appendix B.

## 4. Experiments

**Tasks and Models.** We evaluate ESSA across a broad and diverse set of post-training workloads: (i) School-level math reasoning: train on GSM8K (Cobbe et al., 2021) and report held-out accuracy; backbones: Qwen2.5-7B (Team, 2024) and Qwen2.5-Math-7B (Yang et al., 2024). (ii) Advanced math reasoning: train on PRM800K and evaluate on MATH500 (Lightman et al., 2023), AIME'24/'25 (Zhang & Math-AI, 2024; 2025), MinervaMath (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024) using pass@k and avg@k; backbones: Qwen2.5-Math-7B, Qwen2.5-32B, and Qwen2.5-72B. (iii) Instruction following: train on an ifeval-like-data (Xu et al., 2024b) and evaluate on IFEval (Zhou et al., 2023); backbone: LLaMA-3.1-8B (Grattafiori et al., 2024). (iv) Preference-based assistant behavior: train and evaluate on HelpSteer2 (Wang et al., 2024b) using a reward model; backbone: LLaMA-3.1-8B.

**Baseline.** We use GRPO as the baseline, as it is the standard and most robust online RLHF method for verifiable-reward tasks (Xi et al., 2025; DeepSeek-AI et al., 2025; Yang et al., 2025). Prior works demonstrate that for verifiable-reward tasks like mathematical reasoning, it is standard practice to use RL-based online alignment rather than offline DPO-style methods (Shao et al., 2024a; Lambert et al., 2024). This established separation justifies our focus on GRPO as the relevant and informative baseline for our study. Our goal with ESSA is not to outperform specialized gradient optimizers, but to evaluate whether a gradient-free approach can match the quality of GRPO while operating purely in inference mode.

**SFT.** We start from SFT checkpoints appropriate for each dataset. Experiments without SFT are provided in Appendix F.1. We do not include the time spent on the SFT in all the presented results. It is presented separately in Appendix I.2. Since ESSA and GRPO use the same SFT checkpoint within each backbone-task pair, the SFT cost contributes an identical constant to both methods and does not affect their relative performance. The SFT subset and the online-alignment subset are disjoint splits of the training data (Appendix I).

**Other Details.** We use standard BFLOAT16 precision unless otherwise specified. Qwen2.5-72B is trained under INT4 for per-device evaluation in ESSA. For training the models with the GRPO algorithm, we use the verl library (Sheng et al., 2024), which the authors describe as the most efficient in terms of model allocation and interaction speed.

*Table 2.* ESSA validation maximum accuracy as a function of **GSM8k** dataset fraction used to initialize **Qwen2.5-Math-7B** LoRA. Settings: LoRA rank 16, pop. 192, batch size 100, $\alpha = 1.0$).

| SFT DATA FRACTION | 5% | 25% | 50% | 75% | 100% |
|---|---|---|---|---|---|
| MAX ESSA ACCURACY | 0.713 | 0.731 | 0.807 | 0.863 | 0.872 |

All experiments, unless otherwise noted, are conducted on 8 GPUs. For larger models Qwen2.5-32B and Qwen2.5-72B we use 16 and 32 GPUs, respectively. The sizes of the training and validation datasets, as well as some other details of the experiments, are given in Appendix I.

### 4.1. Sensitivity to Hyperparameters

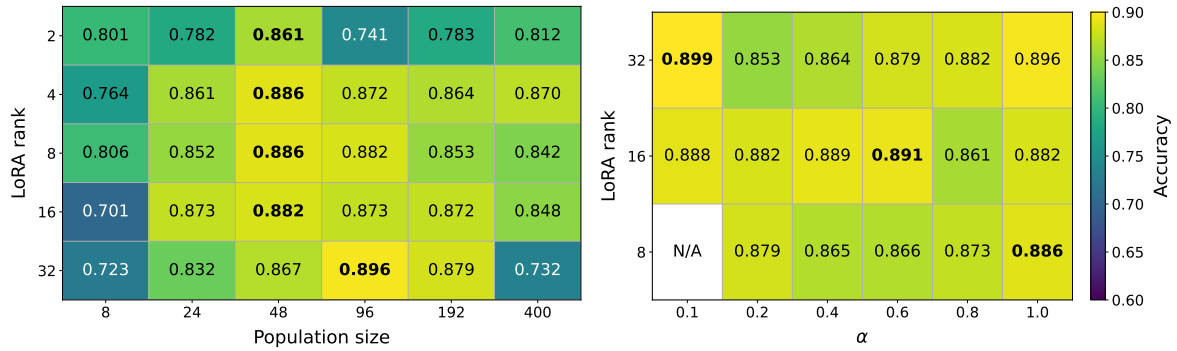
**ESSA hyperparameter sensitivity.** We investigate how ESSA accuracy depends on its key hyperparameters – LoRA rank, population size, and the fraction $\alpha$ of singular values optimized in each SVD factor. A full grid search is performed on five settings Qwen2.5-7B and Qwen2.5-Math-7B on GSM8K and PRM800K, and LLaMA-3.1-8B on IFEval. Figure 2 shows the results for Qwen2.5-Math-7B on GSM8K as a representative example.

We observe three consistent trends: (i) increasing LoRA rank beyond moderate values does not improve accuracy; (ii) very small populations underperform, while gains saturate around 24–96; (iii) accuracy remains stable for $\alpha \geq 0.4$. The complete set of sensitivity plots for the remaining four tasks, which confirm these conclusions and show the cases where rank 2 is optimal, are provided in Appendix C.

**Effect of SFT initialization.** We also examine how the maximum ESSA accuracy depends on the quality of the initial LoRA matrices $A$ and $B$, which are obtained from the SFT stage.

To vary the initialization quality, we traine the SFT model on different fractions of the GSM8K dataset and then run ESSA with identical hyperparameters. The setup is as follows: Qwen2.5-Math-7B, LoRA rank 16, population size 192, and $\alpha = 1.0$. Table 2 reports the final ESSA accuracy as a function of the percentage of SFT data used.

The results show that reducing the SFT dataset from 100% to only 5% lowers the final ESSA accuracy by more than 15 percentage points. Thus, the quality of the initial supervised fine-tuning plays a key role in the ultimate performance of the aligned model. We also conduct an experiment, running the SFT on one domain and training ESSA on another. The results demonstrate ESSA's robustness to the OOD shift in Appendix H.




*(a)* *(b)*

*Figure 2.* Hyperparameter sensitivity of ESSA on **Qwen2.5-Math-7B** for **GSM8K**. Batch size 100. **(a)** Accuracy when varying LoRA rank and population size. **(b)** For each LoRA rank, the population size is fixed to the best value found in (a), while the percentage $\alpha$ of trainable singular values is varied. This illustrates how ESSA performance depends jointly on adapter rank and the fraction of singular values optimized. The single white cell occurs because for LoRA rank 8 and $\alpha = 0.1$, rounding down yields zero trainable singular values, so no valid accuracy is reported.

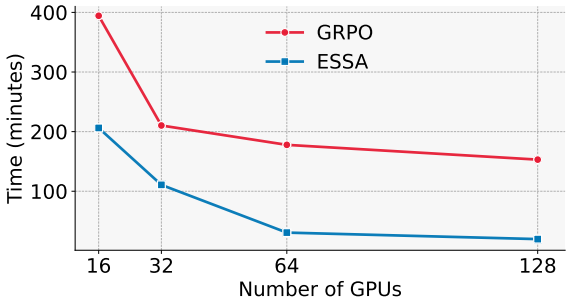



*Figure 3.* GRPO and ESSA scaling on **PRM800K** (evaluated on MATH500) with **Qwen2.5-32B**: time to reach 0.835 accuracy vs. GPU count. ESSA: LoRA rank 16, pop. 128, batch size 256, $\alpha = 1.0$. GRPO: LoRA rank 16, lr $1 \times 10^{-5}$, global batch 512, mini batch 64.

### 4.2. Parallelization

Figure 3 compares the wall-clock time required by GRPO and ESSA to reach a fixed accuracy of 0.835 on Qwen2.5-32B trained on PRM800K for the MATH500 benchmark, when training is distributed across an increasing number of GPUs. For ESSA we use a LoRA rank 16 and an population size 128. For GRPO we match the LoRA rank (16) and use a learning rate $1 \times 10^{-5}$ and global batch size 512.

As the number of GPUs grows from 16 to 128, both methods benefit from additional parallelism, but the gains differ substantially: GRPO decreases from nearly 400 minutes at 16 GPUs to roughly 150 minutes at 128 GPUs ($\approx 2.6\times$ speed-up); ESSA drops from about 200 minutes to under 20 minutes over the same range ($\approx 10\times$ speed-up). On the smallest cluster (16 GPUs) ESSA is already about twice as fast as GRPO. This gap grows with increasing parallelism and reaches a factor of roughly six on 128 GPUs.

ESSA's scaling advantage follows directly from its inference-only optimization loop: evaluation of population members is embarrassingly parallel and requires communication of only random seeds and scalar rewards. In contrast, GRPO must synchronize large gradient tensors each step, leading to communication bottlenecks that limit scaling efficiency.

We also conduct an experiment on a single GPU using the smaller Qwen2.5-3B model, where no parallelism is available. The result is presented in Appendix F.1. This experiment demonstrate that despite the limited hardware ESSA converges faster than GRPO in this setting as well, while the primary goal of this work is to introduce a method that utilizes multi-GPU resources more efficiently than gradient-based training.

### 4.3. Precision Analysis

*Table 3.* ESSA validation maximum accuracy as a function of **Qwen2.5-32B** weight precision on **PRM800K** (validation on MATH500). Settings: LoRA rank 8, pop. 64, $\alpha = 1.0$, batch size 256.

| **Precision** | **BFLOAT16** | **INT8** | **INT4** |
|---|---|---|---|
| Max ESSA accuracy | 0.847 | 0.844 | 0.838 |

Because ESSA uses the model purely in inference mode, it can be trained even when the underlying model weights are quantized. We evaluate this capability on Qwen2.5-32B trained on PRM800K and evaluated on MATH500 with LoRA rank 8, population size 64 and $\alpha = 1.0$. Table 3 reports the best validation accuracy for three numerical precisions. Full training curves are provided in Appendix D.

Accuracy drops by less than one percentage point from BFLOAT16 to INT4. This enables substantial savings in

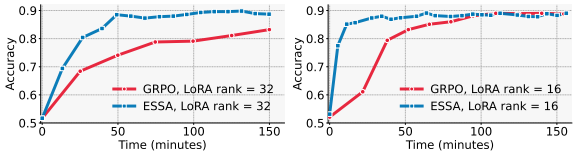
compute and memory: with INT4 quantization, a model as large as 72B parameters can fit on a single GPU for processing each population member, as demonstrated in our large-scale experiments later in the paper.

### 4.4. Comparison to Baseline (GRPO)

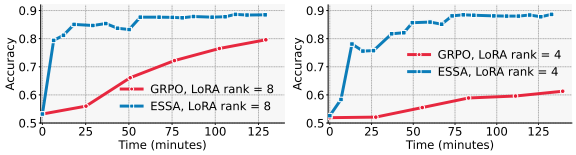
#### 4.4.1. School Math

Figure 4 shows the validation accuracy versus wall-clock time for Qwen2.5-Math-7B on the GSM8K benchmark, comparing ESSA with GRPO across different LoRA ranks. Across all ranks, ESSA rises sharply during the first 10-20 minutes and reaches accuracies near 0.85-0.90 significantly earlier than GRPO, which typically requires 60-100 minutes to approach the same level.

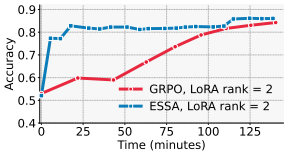
For moderate and large ranks (32, 16) both methods eventually converge to a similar final accuracy, but ESSA attains this plateau far sooner and with less fluctuation. At smaller ranks ESSA maintains accuracy close to its high-rank plateau, while GRPO lags behind for most of training.




*(a)* Population 96, $\alpha = 0.1$ *(b)* Population 48, $\alpha = 0.6$




*(c)* Population 48, $\alpha = 1.0$ *(d)* Population 48, $\alpha = 1.0$




*(e)* Population 48, $\alpha = 1.0$

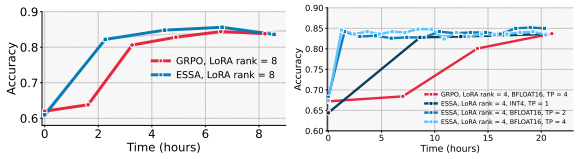
*Figure 4.* Validation accuracy over time on **GSM8K** with **Qwen2.5-Math-7B**. Panels (a)-(e) correspond to LoRA ranks 32, 16, 8, 4, and 2, respectively. ESSA (blue): batch size 100. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64. ESSA rises rapidly and plateaus early across all ranks, while GRPO improves more gradually.

The trajectories demonstrate that ESSA is considerably less sensitive to LoRA rank: lowering the rank from 32 to 2 has only a mild effect on both early-time growth and final accuracy, whereas GRPO's convergence speed degrades markedly as rank decreases. On school-level math reasoning tasks, ESSA consistently delivers a faster time-to-quality than GRPO across all LoRA ranks, making it well suited for rapid iteration or training under tight time budgets. For completeness, the same comparison performed with the Qwen2.5-7B model on GSM8K is reported in Appendix F.1. ESSA consistently converges faster than GRPO while matching final accuracy.

#### 4.4.2. Beyond School Math

Advanced-math benchmarks require multi-step symbolic reasoning, long derivations, and careful numeric manipulation. To faithfully measure the benefit of alignment strategies we therefore employ large backbones, Qwen2.5-32B and Qwen2.5-72B, which (i) possess enough raw capacity to tackle these high-difficulty problems and (ii) let us observe how training methods scale with model size.

Figure 5 illustrates the MATH500 accuracy over time for Qwen2.5-32B and Qwen2.5-72B, respectively. Across both models the curves show the same pattern as on school-math: ESSA rises sharply early, reaches its plateau considerably sooner than GRPO, and maintains that level with low variance.




*(a)* Population 48, $\alpha = 1.0$ *(b)* Population 64, $\alpha = 1.0$

*Figure 5.* MATH500 accuracy over time for models trained on **PRM800K**. **Qwen2.5-32B** with LoRA rank 8 for both methods **(a)** and **Qwen2.5-72B** with LoRA rank 4 for both methods **(b)**. For Qwen2.5-72B we run ESSA under BFLOAT16 with tensor parallelism ($TP$): $TP = 2$ and $TP = 4$, and under INT4 with $TP = 1$, keeping the total GPU budget at 32 for both methods. ESSA (blue): batch size 256. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64. Across both scales, ESSA reaches strong validation accuracy earlier and matches or exceeds GRPO throughout.

In total, we allocate 16 and 32 GPUs to the 32B and 72B models, respectively, across all configurations. In the BFLOAT16 precision regime the 72B model no longer fits on a single GPU, so we use tensor parallelism ($TP$) with size of 2 and 4. GRPO requires at least 4 GPUs for this configuration because both forward and backward passes must be distributed. In contrast, ESSA can operate with INT4 weights, allowing a full 72B model instance to reside on a single GPU and enabling one-candidate-per-GPU evaluation.

We observe that the BFLOAT16 model converges faster than the INT4 version. This happens because placing a 72B model in an INT4 representation leaves less space for KV-cache and INT4 matrix multiplications are slower on standard accelerators. Crucially, increasing tensor parallelism shows minimal impact on the accuracy growth tra-

*Table 4.* Results on advanced-math benchmarks with **Qwen2.5-32B**. Rows are grouped by metric: avg@k and pass@8. For **AIME'24**, **AIME'25**, and **AMC'23**, avg@k is reported at $k = 16$; for other benchmarks, $k = 8$. The final **Avg** averages across available entries.

| Method | MATH500 | MinervaMath | OlympiadBench | AIME'24 | AIME'25 | AMC'23 | Avg |
|---|---|---|---|---|---|---|---|
| | | | **avg@k** | | | | |
| GRPO | 81.8 | 41.2 | 45.7 | 14.6 | 10.0 | 61.7 | 42.5 |
| ESSA | **82.1** | **41.8** | **47.6** | **17.3** | **12.1** | **63.3** | **44.0** |
| | | | **pass@8** | | | | |
| GRPO | **93.6** | **53.3** | 65.3 | 32.9 | **29.2** | **87.3** | 60.2 |
| ESSA | 92.6 | 52.2 | **66.4** | **35.3** | 28.0 | 86.9 | 60.2 |

*Table 5.* Results on advanced-math benchmarks with **Qwen2.5-72B**. Rows are grouped by metric: avg@k and pass@8. For **AIME'24**, **AIME'25**, and **AMC'23**, avg@k is reported at $k = 16$; for other benchmarks, $k = 8$. The final **Avg** averages across available entries.

| Method | MATH500 | MinervaMath | OlympiadBench | AIME'24 | AIME'25 | AMC'23 | Avg |
|---|---|---|---|---|---|---|---|
| | | | **avg@k** | | | | |
| GRPO | **83.0** | 43.0 | **49.1** | **18.8** | 11.0 | 62.2 | 44.5 |
| ESSA | 82.8 | **43.5** | 48.4 | 17.9 | **13.3** | **62.5** | **44.7** |
| | | | **pass@8** | | | | |
| GRPO | **94.2** | 51.8 | 66.7 | **37.5** | 23.5 | 85.6 | 59.9 |
| ESSA | 93.0 | **52.9** | **66.8** | 37.4 | **26.8** | **85.8** | **60.5** |

jectory in this experiment. This indicates that, even when the inference worker does not fit on a single device, that is, when the per-device evaluation rule is broken, ESSA remains efficient.

We also evaluate the trained models on the advanced-math benchmarks using both ESSA and GRPO. For each problem we sample $k$ independent solutions using the same decoding settings and report pass@k (fraction of problems with at least one correct sample) and avg@k (mean correctness averaged over the $k$ samples), following common practice in math-reasoning evaluation. The detailed scores are reported in Table 4 for Qwen2.5-32B and in Table 5 for Qwen2.5-72B. For Qwen2.5-32B, ESSA improves or matches GRPO on avg@k across all benchmarks and shows comparable pass@8 and avg@k performance. For Qwen2.5-72B, where the tasks are especially challenging, ESSA again achieves competitive or better results: it delivers slightly higher pass@8 and avg@k.

### 4.4.3. Instruction Following

We next evaluate ESSA in a domain that is qualitatively different from mathematics: instruction following. Here the goal is to align the model to follow natural language instructions rather than to perform structured reasoning. We use the IFEval benchmark with LLaMA3.1-8B and fix the LoRA rank to 8, comparing ESSA and GRPO under identical data and initialization.

Figure 6 shows that ESSA reaches roughly 0.6 accuracy within the first 60 minutes and maintains that level for the remainder of training. GRPO, in contrast, exhibits a much slower and steadier increase, saturating around 0.45 even after more than four hours. Although the ESSA trajectory displays higher short-term variance, its early and sustained advantage demonstrates that gradient-free evolutionary optimization is effective even in open-ended, non-mathematical instruction-following tasks.

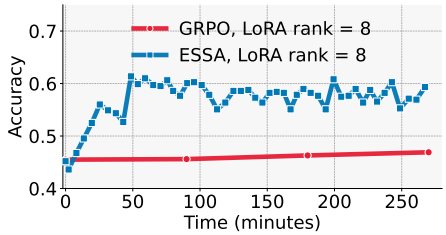



*Figure 6.* Validation accuracy over time on **IFEval** with **LLaMA-3.1-8B.** ESSA (blue): LoRA rank 8, pop. 24, batch size 500, $\alpha = 1.0$. GRPO (red): LoRA rank 8, lr $1 \times 10^{-5}$, global batch 512, mini batch 64. ESSA improves around 0.6-0.65, while GRPO remains nearly flat near 0.45 throughout training.

### 4.4.4. General-Purpose Assistant Setup

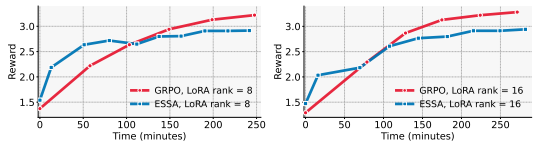
To evaluate alignment in a general-purpose assistant scenario, we employ a more sophisticated reward model rather than simple verifiable rewards. Specifically, we use the RLHFlow/ArmoRM-Llama3-8B-v0.1(Wang et al., 2024a)[1] reward model to provide nuanced, preference-based feedback signals.

The instruction prompts for this setting were drawn from the HelpSteer2 (Wang et al., 2024b) dataset, which contains diverse user instructions and serves as a strong benchmark for open-domain assistant alignment. ESSA and GRPO are both trained with this preference-based reward signal and evaluated under the same protocol to assess their ability to

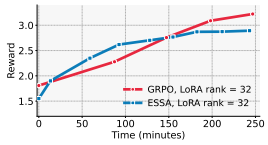
[1] https://huggingface.co/RLHFlow/ArmoRM-Llama3-8B-v0.1

align large language models to general-purpose assistant behavior.

Figure 7 reports the mean per-prompt reward from RLHFlow/ArmoRM-Llama3-8B-v0.1 on the validation set as a function of wall-clock time for LoRA ranks 8, 16, and 32. Across all ranks, ESSA and GRPO achieve similar final reward levels, with their learning curves showing comparable overall trends.




*(a)* Population 36, $\alpha = 1.0$ *(b)* Population 42, $\alpha = 1.0$




*(c)* Population 36, $\alpha = 1.0$

*Figure 7.* Validation reward over time on **HelpSteer2** with **LLaMA-3.1-8B.** Panels (a)-(c) correspond to LoRA ranks 8, 16, 32 respectively. ESSA (blue): batch size 100. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64. Both methods improve steadily, with ESSA showing faster early gains and GRPO slightly higher final reward. Overall, the plots indicate that ESSA achieves comparable alignment quality to GRPO with lower training complexity and similar convergence behavior.

#### 4.4.5. SVD-LoRA in GRPO

In all previous experiments the LoRA factors $A$ and $B$ were fully trainable in GRPO, whereas ESSA optimized only the singular values after an SVD decomposition. To make the comparison strictly fair we repeat the GRPO baseline with the same restriction: only the singular values of the SVD of $A$ and $B$ are updated while their singular vectors were kept fixed, so that both methods have exactly the same number of trainable parameters. This SVD-GRPO variant is evaluated on Qwen2.5-7B with PRM800K, sweeping LoRA ranks 16,8,4,2 and keeping all other GRPO hyperparameters and the SFT initialization identical to the main baseline.

Figure 8 shows that SVD-GRPO struggles to learn effectively: even at rank 16 it plateaus around 0.5 accuracy and degrades further as the rank decreases. By contrast, ESSA with only rank 2 rapidly reaches about 0.72 accuracy and remains stable, outperforming SVD-GRPO by more than twenty percentage points despite operating in an equally low-dimensional parameter space.

This phenomenon can be explained. Deep networks typically exhibit dense, rotated curvature of the objective function, but in high-dimensional parameterizations such as full LoRA, these interactions are spread across many weakly

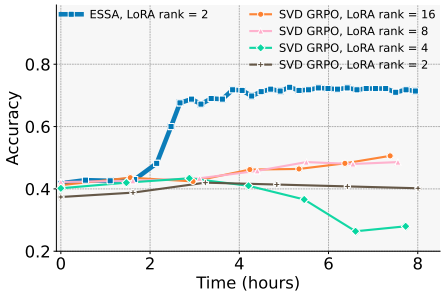



*Figure 8.* MATH500 accuracy over time on **PRM800K** with **Qwen2.5-7B**. ESSA (blue): LoRA rank 2, pop. 24, batch size 300, $\alpha = 1.0$. SVD-GRPO (only singular values are updated): LoRA ranks 16/8/4/2, lr $1 \times 10^{-2}$, global batch 512, mini batch 64. ESSA achieves $\approx 0.72$ accuracy while SVD-GRPO saturates at or below 0.5 even at LoRA rank 16.

correlated coordinates. Each parameter sees only a small fraction of the curvature. After SVD compression, the same curvature is concentrated into a very small subspace, and diagonal first-order optimizers like Adam fail to make progress. ES, by contrast, never computes gradients, its updates depend only on scalar rewards. Moreover CMA-ES adapts its covariance matrix in a way that aligns, in expectation, with the inverse Hessian of objective function.

Thus, for ESSA, moving from direct LoRA parameters to SVD-LoRA simply reduces dimensionality without making the search problem harder. For GRPO, however, the same reparameterization changes both the curvature and the gradient statistics in a way that standard first-order updates are not well adapted to.

## 5. Discussion

ESSA enables scalable LLM alignment without gradients, using inference-only evolutionary search in a compact parameter space. Across the benchmarks studied, ESSA matches and in several cases exceeds GRPO's final quality, while often reaching a given quality level substantially faster in wall-clock time and reducing engineering complexity.

Our theoretical analysis further supports these empirical findings, demonstrating that ESSA's iteration time and parallel efficiency scale substantially better with model and cluster size, thanks to minimal synchronization and communication overhead. The ability to operate natively in low-precision (INT8/INT4) mode enables alignment of very large models – up to 72B parameters – using only a single GPU per candidate, with ¡1 percentage point drop in the best MATH500 accuracy from BF16 to INT4 in our PRM800K 32B experiment (Table 3). These results position evolutionary strategies, when paired with parameter-efficient adaptation, as a compelling alternative to classic RLHF pipelines – offering a simple, scalable, and broadly applicable framework for LLM alignment.

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

## References

Ahmadian, A., Cremer, C., Gallé, M., Fadaee, M., Kreutzer, J., Pietquin, O., Üstün, A., and Hooker, S. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024. URL https://arxiv.org/abs/2402.14740.

Bershatsky, D., Cherniuk, D., Daulbaev, T., Mikhalev, A., and Oseledets, I. Lotr: Low tensor rank weight adaptation, 2024. URL https://arxiv.org/abs/2402.01376.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Bao, H., Xu, H., Wang, H., Ding, H., Xin, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Wang, J., Chen, J., Yuan, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Ye, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Zhao, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., Gregerson, A., Spataru, A., Roziere, B., Biron, B., Tang, B., Chern, B., Caucheteux, C., Nayak, C., Bi, C., Marra, C., McConnell, C., Keller, C., Touret, C., Wu, C., Wong, C., Ferrer, C. C., Nikolaidis, C., Allonsius, D., Song, D., Pintz, D., Livshits, D., Wyatt, D., Esiobu, D., Choudhary, D., Mahajan, D., Garcia-Olano, D., Perino, D., Hupkes, D., Lakomkin, E., AlBadawy, E., Lobanova, E., Dinan, E., Smith, E. M., Radenovic, F., Guzmán, F., Zhang, F., Synnaeve, G., Lee, G., Anderson, G. L., Thattai, G., Nail, G., Mialon, G., Pang, G., Cucurell, G., Nguyen, H., Korevaar, H., Xu, H., Touvron, H., Zarov, I., Ibarra, I. A., Kloumann, I., Misra, I., Evtimov, I., Zhang, J., Copet, J., Lee, J., Geffert, J., Vranes, J., Park, J., Mahadeokar, J., Shah, J., van der Linde, J., Billock, J., Hong, J., Lee, J., Fu, J., Chi, J., Huang, J., Liu, J., Wang, J., Yu, J., Bitton, J., Spisak, J., Park, J., Rocca, J., Johnstun, J., Saxe, J., Jia, J., Alwala, K. V., Prasad, K., Upasani, K., Plawiak, K., Li, K., Heafield, K., Stone, K., El-Arini, K., Iyer, K., Malik, K., Chiu, K., Bhalla, K., Lakhotia, K., Rantala-Yeary, L., van der Maaten, L., Chen, L., Tan, L., Jenkins, L., Martin, L., Madaan, L., Malo, L., Blecher, L., Landzaat, L., de Oliveira, L., Muzzi, M., Pasupuleti, M., Singh, M., Paluri, M., Kardas, M., Tsimpoukelli, M., Oldham, M., Rita, M., Pavlova, M., Kambadur, M., Lewis, M., Si, M., Singh, M. K., Hassan, M., Goyal, N., Torabi, N., Bashlykov, N., Bogoychev, N., Chatterji, N., Zhang, N., Duchenne, O., Çelebi, O., Alrassy, P., Zhang, P., Li, P., Vasic, P., Weng, P., Bhargava, P., Dubal, P., Krishnan, P., Koura, P. S., Xu, P., He, Q., Dong, Q., Srinivasan, R., Ganapathy, R., Calderer, R., Cabral, R. S., Stojnic, R., Raileanu, R., Maheswari, R., Girdhar, R., Patel, R., Sauvestre, R., Polidoro, R., Sumbaly, R., Taylor, R., Silva, R., Hou, R., Wang, R., Hosseini, S., Chennabasappa, S., Singh, S., Bell, S., Kim, S. S., Edunov, S., Nie, S., Narang, S., Raparthy, S., Shen, S., Wan, S., Bhosale, S., Zhang, S., Vandenhende, S., Batra, S., Whitman, S., Sootla, S., Collot, S., Gururangan, S., Borodinsky, S., Herman, T., Fowler, T., Sheasha, T., Georgiou, T., Scialom, T., Speckbacher, T., Mihaylov, T., Xiao, T., Karn, U., Goswami, V., Gupta, V., Ramanathan, V., Kerkez, V., Gonguet, V., Do, V., Vogeti, V., Albiero, V., Petrovic, V., Chu, W., Xiong, W., Fu, W., Meers, W., Martinet, X., Wang, X., Wang,

X., Tan, X. E., Xia, X., Xie, X., Jia, X., Wang, X., Goldschlag, Y., Gaur, Y., Babaei, Y., Wen, Y., Song, Y., Zhang, Y., Li, Y., Mao, Y., Coudert, Z. D., Yan, Z., Chen, Z., Papakipos, Z., Singh, A., Srivastava, A., Jain, A., Kelsey, A., Shajnfeld, A., Gangidi, A., Victoria, A., Goldstand, A., Menon, A., Sharma, A., Boesenberg, A., Baevski, A., Feinstein, A., Kallet, A., Sangani, A., Teo, A., Yunus, A., Lupu, A., Alvarado, A., Caples, A., Gu, A., Ho, A., Poulton, A., Ryan, A., Ramchandani, A., Dong, A., Franco, A., Goyal, A., Saraf, A., Chowdhury, A., Gabriel, A., Bharambe, A., Eisenman, A., Yazdan, A., James, B., Maurer, B., Leonhardi, B., Huang, B., Loyd, B., Paola, B. D., Paranjape, B., Liu, B., Wu, B., Ni, B., Hancock, B., Wasti, B., Spence, B., Stojkovic, B., Gamido, B., Montalvo, B., Parker, C., Burton, C., Mejia, C., Liu, C., Wang, C., Kim, C., Zhou, C., Hu, C., Chu, C.-H., Cai, C., Tindal, C., Feichtenhofer, C., Gao, C., Civin, D., Beaty, D., Kreymer, D., Li, D., Adkins, D., Xu, D., Testuggine, D., David, D., Parikh, D., Liskovich, D., Foss, D., Wang, D., Le, D., Holland, D., Dowling, E., Jamil, E., Montgomery, E., Presani, E., Hahn, E., Wood, E., Le, E.-T., Brinkman, E., Arcaute, E., Dunbar, E., Smothers, E., Sun, F., Kreuk, F., Tian, F., Kokkinos, F., Ozgenel, F., Caggioni, F., Kanayet, F., Seide, F., Florez, G. M., Schwarz, G., Badeer, G., Swee, G., Halpern, G., Herman, G., Sizov, G., Guangyi, Zhang, Lakshminarayanan, G., Inan, H., Shojanazeri, H., Zou, H., Wang, H., Zha, H., Habeeb, H., Rudolph, H., Suk, H., Aspegren, H., Goldman, H., Zhan, H., Damlaj, I., Molybog, I., Tufanov, I., Leontiadis, I., Veliche, I.-E., Gat, I., Weissman, J., Geboski, J., Kohli, J., Lam, J., Asher, J., Gaya, J.-B., Marcus, J., Tang, J., Chan, J., Zhen, J., Reizenstein, J., Teboul, J., Zhong, J., Jin, J., Yang, J., Cummings, J., Carvill, J., Shepard, J., McPhie, J., Torres, J., Ginsburg, J., Wang, J., Wu, K., U, K. H., Saxena, K., Khandelwal, K., Zand, K., Matosich, K., Veeraraghavan, K., Michelena, K., Li, K., Jagadeesh, K., Huang, K., Chawla, K., Huang, K., Chen, L., Garg, L., A, L., Silva, L., Bell, L., Zhang, L., Guo, L., Yu, L., Moshkovich, L., Wehrstedt, L., Khabsa, M., Avalani, M., Bhatt, M., Mankus, M., Hasson, M., Lennie, M., Reso, M., Groshev, M., Naumov, M., Lathi, M., Keneally, M., Liu, M., Seltzer, M. L., Valko, M., Restrepo, M., Patel, M., Vyatskov, M., Samvelyan, M., Clark, M., Macey, M., Wang, M., Hermoso, M. J., Metanat, M., Rastegari, M., Bansal, M., Santhanam, N., Parks, N., White, N., Bawa, N., Singhal, N., Egebo, N., Usunier, N., Mehta, N., Laptev, N. P., Dong, N., Cheng, N., Chernoguz, O., Hart, O., Salpekar, O., Kalinli, O., Kent, P., Parekh, P., Saab, P., Balaji, P., Rittner, P., Bontrager, P., Roux, P., Dollar, P., Zvyagina, P., Ratanchandani, P., Yuvraj, P., Liang, Q., Alao, R., Rodriguez, R., Ayub, R., Murthy, R., Nayani, R., Mitra, R., Parthasarathy, R., Li, R., Hogan, R., Battey, R., Wang, R., Howes, R., Rinott, R., Mehta, S., Siby, S., Bondu, S. J., Datta, S., Chugh, S., Hunt, S., Dhillon, S., Sidorov, S., Pan, S., Mahajan, S., Verma, S., Yamamoto, S., Ramaswamy, S., Lindsay, S., Lindsay, S., Feng, S., Lin, S., Zha, S. C., Patil, S., Shankar, S., Zhang, S., Zhang, S., Wang, S., Agarwal, S., Sajuyigbe, S., Chintala, S., Max, S., Chen, S., Kehoe, S., Satterfield, S., Govindaprasad, S., Gupta, S., Deng, S., Cho, S., Virk, S., Subramanian, S., Choudhury, S., Goldman, S., Remez, T., Glaser, T., Best, T., Koehler, T., Robinson, T., Li, T., Zhang, T., Matthews, T., Chou, T., Shaked, T., Vontimitta, V., Ajayi, V., Montanez, V., Mohan, V., Kumar, V. S., Mangla, V., Ionescu, V., Poenaru, V., Mihailescu, V. T., Ivanov, V., Li, W., Wang, W., Jiang, W., Bouaziz, W., Constable, W., Tang, X., Wu, X., Wang, X., Wu, X., Gao, X., Kleinman, Y., Chen, Y., Hu, Y., Jia, Y., Qi, Y., Li, Y., Zhang, Y., Zhang, Y., Adi, Y., Nam, Y., Yu, Wang, Zhao, Y., Hao, Y., Qian, Y., Li, Y., He, Y., Rait, Z., DeVito, Z., Rosnbrick, Z., Wen, Z., Yang, Z., Zhao, Z., and Ma, Z. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Hansen, N. and Ostermeier, A. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z. L., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

Hong, J., Lee, N., and Thorne, J. Orpo: Monolithic preference optimization without reference model, 2024. URL https://arxiv.org/abs/2403.07691.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp, 2019. URL https://arxiv.org/abs/1902.00751.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Hu, J. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

Huang, C., Liu, Q., Lin, B. Y., Pang, T., Du, C., and Lin, M. Lorahub: Efficient cross-task generalization via dynamic lora composition, 2024. URL https://arxiv.org/abs/2307.13269.

Jin, F., Liu, Y., and Tan, Y. Derivative-free optimization for low-rank adaptation in large language models, 2024. URL https://arxiv.org/abs/2403.01754.

Kopiczko, D. J., Blankevoort, T., and Asano, Y. M. Vera: Vector-based random matrix adaptation, 2024. URL https://arxiv.org/abs/2310.11454.

Lambert, N., Morrison, J., Pyatkin, V., Huang, S., Ivison, H., Brahman, F., Miranda, L. J. V., Liu, A., Dziri, N., Lyu, S., et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

LeCun, Y. and Cortes, C. The mnist database of handwritten digits. 2005. URL https://api.semanticscholar.org/CorpusID:60282629.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation, 2021. URL https://arxiv.org/abs/2101.00190.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H. Dora: Weight-decomposed low-rank adaptation, 2024. URL https://arxiv.org/abs/2402.09353.

Maheswaranathan, N., Metz, L., Choi, D., Tucker, G., and Sohl-Dickstein, J. Guided evolutionary strategies: escaping the curse of dimensionality in random search. 2018. URL https://arxiv.org/abs/1806.10230.

Mania, H., Guy, A., and Recht, B. Simple random search provides a competitive approach to reinforcement learning, 2018. URL https://arxiv.org/abs/1803.07055.

Meng, Y., Xia, M., and Chen, D. Simpo: Simple preference optimization with a reference-free reward, 2024. URL https://arxiv.org/abs/2405.14734.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022a. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback, 2022b. URL https://arxiv.org/abs/2203.02155.

Qiu, X., Gan, Y., Hayes, C. F., Liang, Q., Meyerson, E., Hodjat, B., and Miikkulainen, R. Evolution strategies at scale: Llm fine-tuning beyond reinforcement learning, 2025. URL https://arxiv.org/abs/2509.24372.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

Salimans, T., Ho, J., Chen, X., Sidor, S., and Sutskever, I. Evolution strategies as a scalable alternative to reinforcement learning, 2017. URL https://arxiv.org/abs/1703.03864.

Sarkar, B., Fellows, M., Duque, J. A., Letcher, A., Villares, A. L., Sims, A., Cope, D., Liesen, J., Seier, L., Wolf, T., Berdica, U., Goldie, A. D., Courville, A., Sevegnani, K., Whiteson, S., and Foerster, J. N. Evolution strategies at the hyperscale, 2025. URL https://arxiv.org/abs/2511.16652.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024a.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y., and Guo, D. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024b. URL https://arxiv.org/abs/2402.03300.

Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.

Sun, Q., Cetin, E., and Tang, Y. Transformer-squared: Self-adaptive llms, 2025. URL https://arxiv.org/abs/2501.06252.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In Solla, S., Leen, T., and Müller, K. (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

Tang, Y., Guo, D. Z., Zheng, Z., Calandriello, D., Cao, Y., Tarassov, E., Munos, R., Ávila Pires, B., Valko, M., Cheng, Y., and Dabney, W. Understanding the performance gap between online and offline alignment algorithms, 2024. URL https://arxiv.org/abs/2405.08448.

Team, Q. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, H., Xiong, W., Xie, T., Zhao, H., and Zhang, T. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *EMNLP*, 2024a.

Wang, Z., Dong, Y., Delalleau, O., Zeng, J., Shen, G., Egert, D., Zhang, J. J., Sreedhar, M. N., and Kuchaiev, O. Helpsteer2: Open-source dataset for training top-performing reward models, 2024b.

Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., and Schmidhuber, J. Natural evolution strategies, 2011. URL https://arxiv.org/abs/1106.4487.

Xi, Z., Huang, J., Liao, C., Huang, B., Guo, H., Liu, J., Zheng, R., Ye, J., Zhang, J., Chen, W., He, W., Ding, Y., Li, G., Chen, Z., Du, Z., Yao, X., Xu, Y., Chen, J., Gui, T., Wu, Z., Zhang, Q., Huang, X., and Jiang, Y.-G. Agentgym-rl: Training llm agents for long-horizon decision making through multi-turn reinforcement learning, 2025. URL https://arxiv.org/abs/2509.08755.

Xu, S., Fu, W., Gao, J., Ye, W., Liu, W., Mei, Z., Wang, G., Yu, C., and Wu, Y. Is dpo superior to ppo for llm alignment? a comprehensive study, 2024a. URL https://arxiv.org/abs/2404.10719.

Xu, Z., Jiang, F., Niu, L., Deng, Y., Poovendran, R., Choi, Y., and Lin, B. Y. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing, 2024b. URL https://arxiv.org/abs/2406.08464.

Yang, A., Zhang, B., Hui, B., Gao, B., Yu, B., Li, C., Liu, D., Tu, J., Zhou, J., Lin, J., Lu, K., Xue, M., Lin, R., Liu, T., Ren, X., and Zhang, Z. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024. URL https://arxiv.org/abs/2409.12122.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., Liu, X., Lin, H., Lin, Z., Ma, B., Sheng, G., Tong, Y., Zhang, C., Zhang, M., Zhang, W., Zhu, H., Zhu, J., Chen, J., Chen, J., Wang, C., Yu, H., Song, Y., Wei, X., Zhou, H., Liu, J., Ma, W.-Y., Zhang, Y.-Q., Yan, L., Qiao, M., Wu, Y., and Wang, M. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Zhang, Y. and Math-AI, T. American invitational mathematics examination (aime) 2024, 2024.

Zhang, Y. and Math-AI, T. American invitational mathematics examination (aime) 2025, 2025.

Zhang, Y., Li, P., Hong, J., Li, J., Zhang, Y., Zheng, W., Chen, P.-Y., Lee, J. D., Yin, W., Hong, M., Wang, Z., Liu, S., and Chen, T. Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark, 2024. URL https://arxiv.org/abs/2402.11592.

Zhang, Y., Ye, P., Yang, X., Feng, S., Zhang, S., Bai, L., Ouyang, W., and Hu, S. Nature-inspired population-based evolution of large language models, 2025. URL https://arxiv.org/abs/2503.01155.

Zheng, R., Dou, S., Gao, S., Hua, Y., Shen, W., Wang, B., Liu, Y., Jin, S., Liu, Q., Zhou, Y., Xiong, L., Chen, L., Xi, Z., Xu, N., Lai, W., Zhu, M., Chang, C., Yin, Z., Weng, R., Cheng, W., Huang, H., Sun, T., Yan, H., Gui, T., Zhang, Q., Qiu, X., and Huang, X. Secrets of

rlhf in large language models part i: Ppo, 2023. URL https://arxiv.org/abs/2307.04964.

Zhou, J., Lu, T., Mishra, S., Brahma, S., Basu, S., Luan, Y., Zhou, D., and Hou, L. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

**Algorithm 1** ESSA: Distributed Evolutionary Search over LoRA Singular Values

**Require:** Training set $\mathcal{D}$, LoRA factors $A_i \in \mathbb{R}^{r\times n}$, $B_i \in \mathbb{R}^{m\times r}$ for all attention matrices, fraction $\alpha$ of top singular values to optimize, population size $P$, number of GPUs $N$, number of ES generations $E$.

**SVD initialization:**
**for** each LoRA factor $A_i, B_i$ **do**
  $A_i = U_{A_i}\Sigma_{A_i}V_{A_i}^\top$, $B_i = U_{B_i}\Sigma_{B_i}V_{B_i}^\top$
  Keep $U_{A_i}, V_{A_i}, U_{B_i}, V_{B_i}$ fixed; collect singular values $\sigma_{A_i} = \mathrm{diag}(\Sigma_{A_i})$, $\sigma_{B_i} = \mathrm{diag}(\Sigma_{B_i})$
**end for**
Concatenate the top $\alpha \times 100\%$ singular values of each $\sigma_{A_i}, \sigma_{B_i}$ into a single parameter vector $\sigma$.
Initialize $N$ workers with identical random seeds and a common CMA-ES state.
**for** $t = 0, 1, \ldots, E-1$ **do**
  **for** each worker $k = 1, \ldots, N$ **in parallel do**
    Sample candidate perturbations of $\sigma$ using the shared random seeds.
    For each candidate:

1. Reconstruct $A_i', B_i'$ by adding perturbation vector to $\sigma$ and computing

$$A_i' = U_{A_i}\,\mathrm{diag}(\tilde{\sigma}_{A_i})\,V_{A_i}^\top,\ B_i' = U_{B_i}\,\mathrm{diag}(\tilde{\sigma}_{B_i})\,V_{B_i}^\top.$$

2. Update the model weights $W_i' = W_i + B_i'A_i'$.

3. Draw a random mini-batch $\mathcal{D}_{\text{mini}} \subset \mathcal{D}$ of fixed size and compute the reward $F = F(W'; \mathcal{D}_{\text{mini}})$.

  **end for**
  All workers exchange seeds and the corresponding rewards so that each worker knows the full set of evaluations.
  Each worker reconstructs the same perturbations from the shared seeds and performs one CMA-ES update of the search distribution.
**end for**
Reconstruct the final LoRA factors from the evolved singular values and return the aligned model.

## A. ESSA Algorithm

The overall procedure of ESSA is summarized in Algorithm 1, which outlines the main steps of the distributed evolutionary search over LoRA singular values.

## B. Detailed Theoretical Analysis

In this appendix we compare the iteration latency (wall-clock time per optimization step) of gradient-based online alignment (e.g., PPO/GRPO) and ESSA.

Let $G$ be the number of GPUs. Let $T^{\text{grad}}$ denote the per-iteration time of an idealized gradient-based RLHF pipeline and $T^{\text{essa}}$ the per-iteration time of ESSA. We use the following quantities: $B^{\text{grad}}$ is the global batch size per gradient iteration; $N_{\text{pop}}$ is the ESSA population size (number of candidates evaluated per iteration); $B^{\text{essa}}$ is the per-candidate batch size in ESSA (number of prompts/samples evaluated by a single candidate); $\eta_{\text{gen}}$ is the average generation time per sample (seconds/sample); $\eta_{\text{fb}}$ is the average forward-backward time per sample for the gradient update (seconds/sample); $M_{\text{params}}$ is the size of the communicated gradient tensor (bytes), which equals the model-parameter size; and $\text{peak_bw}$ is the effective per-GPU bandwidth for large collective communication (bytes/second).

Gradient methods incur an all-reduce communication cost proportional to $M_{\text{params}}$, while ESSA exchanges only constant-size information (random seeds and scalar rewards). This yields the following sufficient condition for ESSA to have strictly lower iteration latency.

**Proposition B.1** (ESSA iteration-time advantage)**.** *If the ESSA population size $N_{\text{pop}}$ and per-candidate batch size $B^{\text{essa}}$ satisfy*

$$N_{\text{pop}} B^{\text{essa}} < B^{\text{grad}} \left(1 + \frac{\eta_{\text{fb}}}{G\,\eta_{\text{gen}}}\right) + \frac{(G-1)\,M_{\text{params}}}{\text{peak_bw}\,G\,\eta_{\text{gen}}}, \tag{1}$$

*then the ESSA iteration time is strictly smaller than that of the idealized gradient pipeline:*

$$T^{\text{essa}} < T^{\text{grad}}.$$

To fully prove Proposition B.1, we proceed in several steps. We first introduce the notation and expressions for per-iteration computation and communication time for both gradient-based methods and ESSA, then establish the optimal split of devices between training and generation, and finally prove a conservative bound on the ESSA population size under which ESSA is guaranteed to be faster.

**Notation.** Let $B^{\text{grad}}$ be the global batch size used in gradient methods, $B^{\text{essa}}$ the batch size processed by a single population instance, $b_{\text{fb}}, b_{\text{gen}}$ the microbatch sizes, $m_{\text{fb}}, m_{\text{gen}}$ GPUs per model instance for training and generation, $G$ total GPUs with $G_{\text{fb}}, G_{\text{gen}}$ allocated to each, $\tau_{\text{fb}}(b), \tau_{\text{gen}}(b)$ the forward-backward and generation microbatch times, $\eta_{\text{fb}} = \tau_{\text{fb}}(b_{\text{fb}})/b_{\text{fb}}$ and $\eta_{\text{gen}} = \tau_{\text{gen}}(b_{\text{gen}})/b_{\text{gen}}$ the per-sample times, and $k_{\text{fb}}^{\text{parallel}} = G_{\text{fb}}/m_{\text{fb}}$, $k_{\text{gen}}^{\text{parallel}} = G_{\text{gen}}/m_{\text{gen}}$ the numbers of microbatches processed in parallel. $T^{\text{grad}} = T^{\text{grad}}_{\text{fb-gen}} + T^{\text{grad}}_{\text{comm}}$, $T^{\text{essa}} = T^{\text{essa}}_{\text{gen}} + T^{\text{essa}}_{\text{comm}}$, with asynchronous training and generation so that $T^{\text{grad}}_{\text{fb-gen}} = \max\big(T^{\text{grad}}_{\text{fb}},\ T^{\text{grad}}_{\text{gen}}\big)$.

**Computation.** Processing $B^{\text{grad}}$ samples by gradient training requires $B^{\text{grad}}/b_{\text{fb}}$ microbatches; $k_{\text{fb}}^{\text{parallel}}$ run in parallel:

$$T^{\text{grad}}_{\text{fb}} = \frac{B^{\text{grad}}}{b_{\text{fb}}\,k_{\text{fb}}^{\text{parallel}}}\,\tau_{\text{fb}}(b_{\text{fb}}) = \frac{B^{\text{grad}}\,m_{\text{fb}}}{G_{\text{fb}}}\,\eta_{\text{fb}}. \qquad T^{\text{grad}}_{\text{gen}} = \frac{B^{\text{grad}}\,m_{\text{gen}}}{G_{\text{gen}}}\,\eta_{\text{gen}}.$$

For ESSA, a population of $N_{\text{pop}}$ candidates is evaluated purely by generation:

$$T^{\text{essa}}_{\text{gen}} = \frac{N_{\text{pop}} B^{\text{essa}}\,m_{\text{gen}}}{G}\,\eta_{\text{gen}}.$$

**Communication.** Let $M_{\text{params}}$ be the model-parameter size (bytes). Gradient methods communicate gradients using all-reduce which consists of two collective operations: reduce-scatter and all-gather. Each moves a block of size $M_{\text{params}}/G$ across the $G-1$ other devices. With effective interconnect peak bandwidth $\text{peak_bw}$ (bytes/s) – the sustained per-GPU bandwidth for large collective messages:

$$T^{\text{grad}}_{\text{comm}} = 2 \cdot \frac{M_{\text{params}}(G-1)}{G\,\text{peak_bw}}.$$

ESSA communicates only a random seed and the resulting reward, requiring a single all-gather of size $M_{\text{essa}} = 2 \times 4$ bytes:

$$T^{\text{essa}}_{\text{comm}} = \frac{M_{\text{essa}}(G-1)}{G\,\text{peak_bw}}.$$

**Optimal device split.** Let $\theta \in (0,1)$ be the fraction of devices used for training ($G_{\text{fb}} = \theta G$, $G_{\text{gen}} = (1-\theta)G$). Then

$$T^{\text{grad}}_{\text{fb-gen}}(\theta) = \frac{B^{\text{grad}}}{G}\,\max\left(\frac{m_{\text{fb}}\eta_{\text{fb}}}{\theta}, \frac{m_{\text{gen}}\eta_{\text{gen}}}{1-\theta}\right).$$

**Lemma B.2** (Optimal split)**.** *The minimum of $T^{\text{grad}}_{\text{fb-gen}}(\theta)$ over $\theta \in (0,1)$ is attained at*

$$\theta^\star = \frac{m_{\text{fb}}\eta_{\text{fb}}}{m_{\text{fb}}\eta_{\text{fb}} + m_{\text{gen}}\eta_{\text{gen}}}, \qquad T^{\text{grad}}_{\text{fb-gen}}(\theta^\star) = \frac{B^{\text{grad}}}{G}\big(m_{\text{fb}}\eta_{\text{fb}} + m_{\text{gen}}\eta_{\text{gen}}\big).$$

*Proof.* $\max(a/\theta, b/(1-\theta))$ is minimized when the two arguments are equal: $a/\theta = b/(1-\theta)$, with $a = m_{\text{fb}}\eta_{\text{fb}}$, $b = m_{\text{gen}}\eta_{\text{gen}}$. □

Define the ideal gradient iteration time (perfect scheduling):

$$T_\star^{\mathrm{grad}} = T_{\text{fb-gen}}^{\mathrm{grad}}(\theta^\star) + T_{\mathrm{comm}}^{\mathrm{grad}} = \frac{B^{\mathrm{grad}}}{G}\big(m_{\mathrm{fb}}\eta_{\mathrm{fb}} + m_{\mathrm{gen}}\eta_{\mathrm{gen}}\big) + 2 \cdot \frac{M_{\mathrm{params}}(G-1)}{G\,\mathrm{peak_bw}}.$$

Clearly $T^{\mathrm{grad}} \geq T_\star^{\mathrm{grad}}$.

**Theorem B.3** (ESSA iteration is faster under a conservative bound)**.** *Suppose we only assume* $m_{\mathrm{fb}} \geq 1$, $m_{\mathrm{gen}} \leq G$, *and* $2M_{\mathrm{params}} - M_{\mathrm{essa}} \geq M_{\mathrm{params}}$. *If the population size satisfies*

$$N_{\mathrm{pop}} B^{\mathrm{essa}} \;<\; B^{\mathrm{grad}}\left(\frac{\eta_{\mathrm{fb}}}{G\eta_{\mathrm{gen}}} + 1\right) + \frac{(G-1)M_{\mathrm{params}}}{\mathrm{peak_bw}\; G\eta_{\mathrm{gen}}}, \tag{2}$$

*then* $T^{\mathrm{essa}} < T_\star^{\mathrm{grad}} \leq T^{\mathrm{grad}}$.

*Proof.* Starting from $T^{\mathrm{essa}} < T_\star^{\mathrm{grad}}$ and applying the bounds $m_{\mathrm{fb}} \geq 1$, $m_{\mathrm{gen}} \leq G$, $2M_{\mathrm{params}} - M_{\mathrm{essa}} \geq M_{\mathrm{params}}$ to the exact inequality yields (2). □

**Discussion.** Theorem B.3 provides a conservative population-size threshold below which ESSA is guaranteed to be faster than the idealized gradient pipeline. Because real clusters rarely achieve the perfect split $\theta^\star$ and typically operate below the nominal $\mathrm{peak_bw}$, the practical advantage of ESSA is often even larger than predicted by (2).

# C. Sensitivity to Hyperparameters

## C.1. Heatmaps

For completeness, Figures 9-12 present the full hyperparameter sensitivity for Section 4.1. These include Qwen2.5-7B on both GSM8K and PRM800K, Qwen2.5-Math-7B on PRM800K, as well as LLaMA-3.1-8B on IFEval.

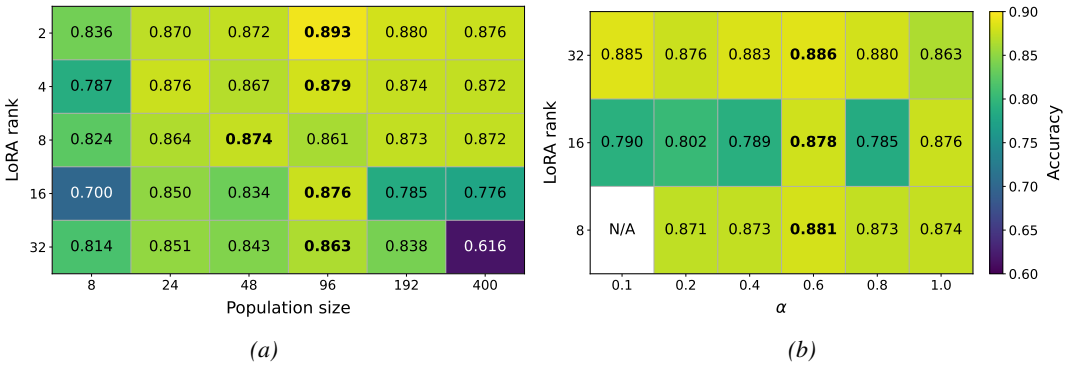



*Figure 9.* Hyperparameter sensitivity of ESSA on **Qwen2.5-7B** for **GSM8K**. Batch size 100. **(a)** Accuracy when varying LoRA rank and population size. **(b)** For each LoRA rank, the population size is fixed to the best value found in (a), while the percentage $\alpha$ of trainable singular values is varied. This illustrates how ESSA performance depends jointly on adapter rank and the fraction of singular values optimized. The single white cell occurs because for LoRA rank 8 and $\alpha = 0.1$, rounding down yields zero trainable singular values, so no valid accuracy is reported.

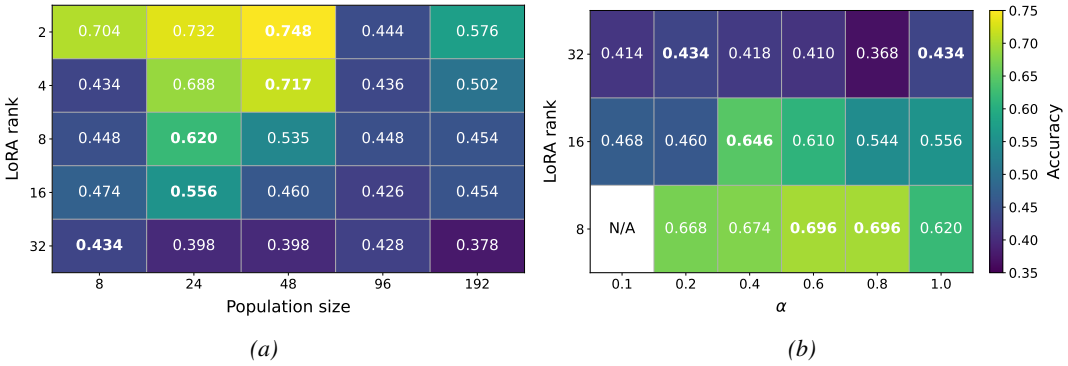



*Figure 10.* Hyperparameter sensitivity of ESSA on **Qwen2.5-7B** for **PRM800K** (evaluated on MATH500). Batch size 300. **(a)** Accuracy when varying LoRA rank and population size. **(b)** For each LoRA rank, the population size is fixed to the best value found in (a), while the percentage $\alpha$ of trainable singular values is varied. This illustrates how ESSA performance depends jointly on adapter rank and the fraction of singular values optimized. The single white cell occurs because for LoRA rank 8 and $\alpha = 0.1$, rounding down yields zero trainable singular values, so no valid accuracy is reported.

Figure 9 examines the hyperparameter sensitivity of ESSA on Qwen2.5-7B for GSM8K. Varying the LoRA rank and population size shows that performance remains stable across a broad range, with accuracy peaking at 0.893 for LoRA rank 2 and population size 96. Notably, even very low-ranks (2-4) achieve top performance, while larger ranks yield diminishing or slightly degraded results. When fixing the population size and varying the fraction $\alpha$ of trainable singular values, ESSA maintains consistently high accuracy ($\approx$ 0.87-0.89) across all $\alpha$ values. This indicates strong robustness to the degree of SVD sparsification.

Figure 10 analyzes the hyperparameter sensitivity of Qwen2.5-7B trained with ESSA on PRM800K and evaluated on MATH500. When varying LoRA rank and population size, accuracy peaks at 0.748 for LoRA rank 2 and population size 48, showing that small adapter ranks with moderate population sizes are most effective. Performance drops notably for larger ranks (16) or oversized populations (96), indicating diminishing returns beyond a compact search space. With population size fixed, varying the fraction $\alpha$ of trainable singular values shows that moderate values ($\alpha \approx$ 0.4-0.8) yield the best results – up to 0.696 accuracy for LoRA rank 8 – while very low or full updates slightly reduce performance. These results confirm that ESSA performs best with low-rank adapters, moderate population sizes, and partial singular value optimization.

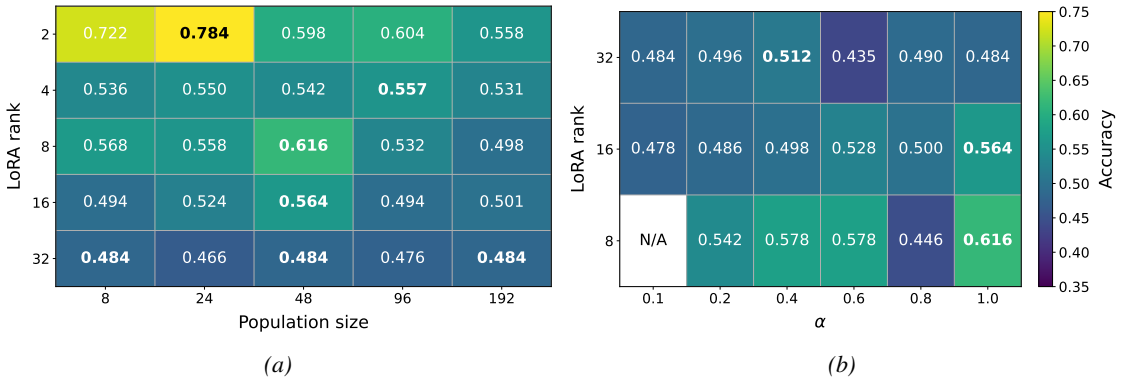



*Figure 11.* Hyperparameter sensitivity of ESSA on **Qwen2.5-Math-7B** for **PRM800K** (evaluated on MATH500). Batch size 300. **(a)** Accuracy when varying LoRA rank and population size. **(b)** For each LoRA rank, the population size is fixed to the best value found in (a), while the percentage $\alpha$ of trainable singular values is varied. This illustrates how ESSA performance depends jointly on adapter rank and the fraction of singular values optimized. The single white cell occurs because for LoRA rank 8 and $\alpha = 0.1$, rounding down yields zero trainable singular values, so no valid accuracy is reported.

Figure 11 presents the hyperparameter sensitivity of Qwen2.5-Math-7B trained with ESSA on PRM800K and evaluated on MATH500. When varying LoRA rank and population size, accuracy peaks at 0.784 for LoRA rank 2 and population size 24, showing that compact low-rank adapters with small populations perform best. Higher ranks or very large populations lead to gradual accuracy degradation, indicating that excessive search dimensionality does not improve performance. Fixing population size and varying the fraction $\alpha$ of trainable singular values, ESSA achieves its highest score (0.616) at LoRA rank 8 and full update ($\alpha = 1.0$), while intermediate $\alpha$ values yield competitive but slightly lower results. The results highlight that ESSA performs optimally with lightweight low-rank configurations and remains robust to SVD sparsification, even on complex reasoning tasks.

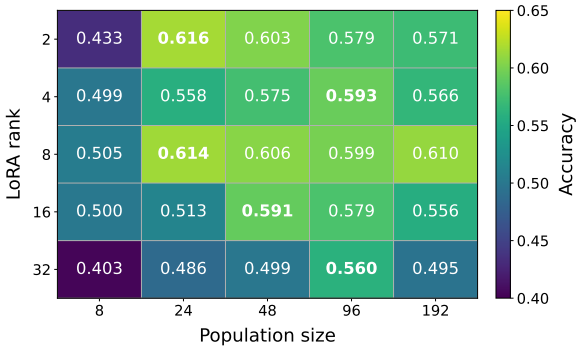


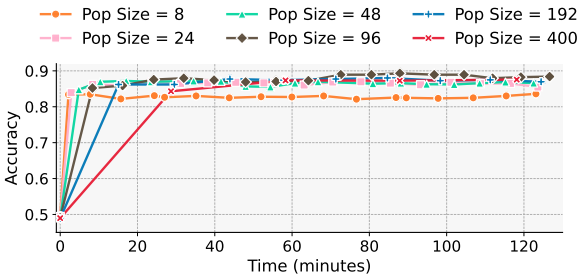

*Figure 12.* Hyperparameter sensitivity of ESSA on **LLaMA-3.1-8B** for **IFeval**. Accuracy when varying LoRA rank and population size.

Figure 12 shows the hyperparameter sensitivity of ESSA on LLaMA-3.1-8B for IFeval. Varying LoRA rank and population size reveals stable performance across configurations, with the highest accuracy (0.614) achieved at LoRA rank 8 and population size 24. Low ranks (4-16) yield comparable results, while very high-rank (32) adapters slightly underperform, suggesting a sweet spot around low-range ranks and moderate population sizes.

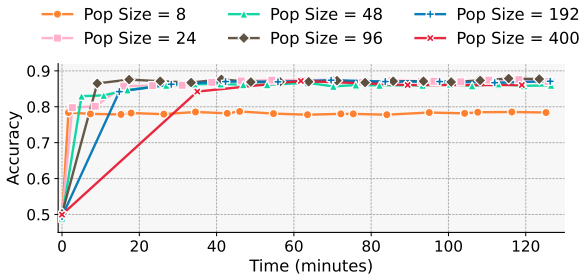
## C.2. Training Dynamics

### C.2.1. Qwen2.5-7B on GSM8K

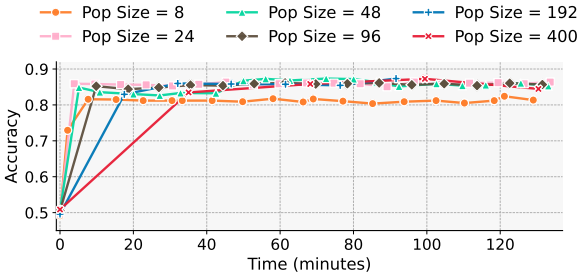
Appendix Figures 13, 14 and 15 provide the complete training-dynamic study of ESSA on Qwen2.5-7B for GSM8K. This analysis shows how validation accuracy evolves over time as we vary the three key ESSA hyperparameters (Figure 9).



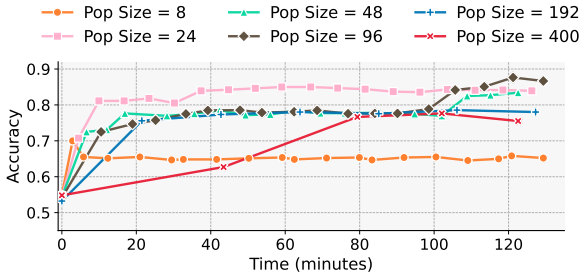

*(a)* LoRA rank 2



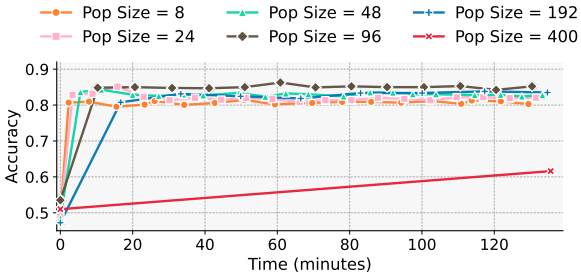

*(b)* LoRA rank 4




*(c)* LoRA rank 8



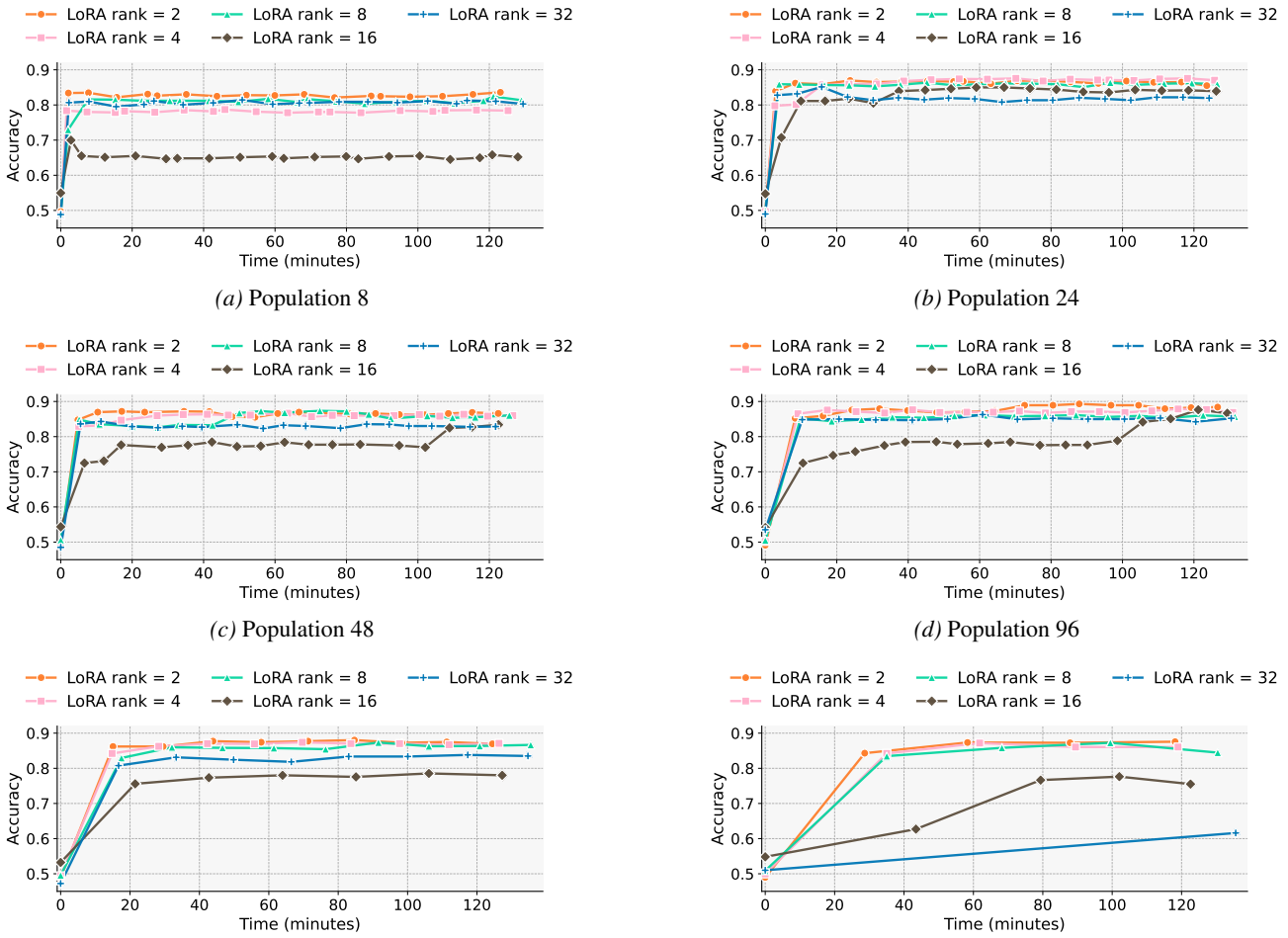

*(d)* LoRA rank 16




*(e)* LoRA rank 32

*Figure 13.* Validation accuracy over time on **GSM8K** with **Qwen2.5-7B** when varying the ESSA population size (8, 24, 48, 96, 192, 400) at fixed LoRA ranks (2, 4, 8, 16, 32). Batch size 100.

Figure 13 shows the full training dynamics of ESSA on GSM8K with Qwen2.5-7B when varying the ESSA population size.




*(a)* Population 8

*(b)* Population 24

*(c)* Population 48

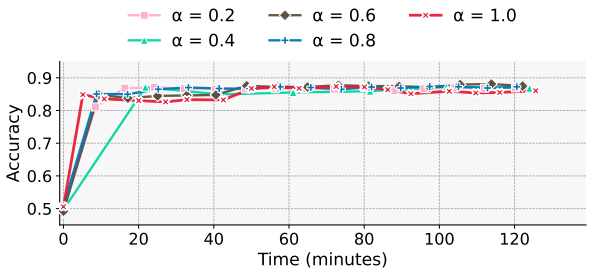
*(d)* Population 96

*(e)* Population 192

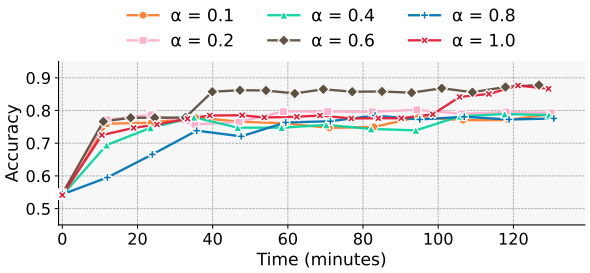
*(f)* Population 400

*Figure 14.* Validation accuracy over time on **GSM8K** with **Qwen2.5-7B** when varying the LoRA rank (2, 4, 8, 16, 32) at fixed population sizes (8, 24, 48, 96, 192, 400). Batch size 100.

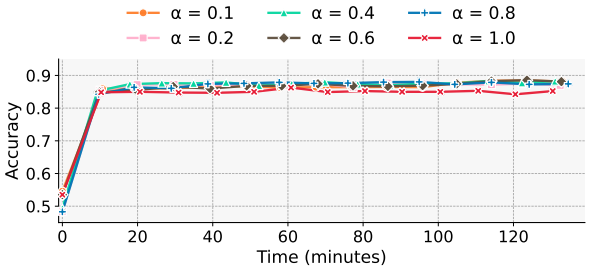
Figure 14 shows the full training dynamics of ESSA on GSM8K with Qwen2.5-7B when varying the LoRA rank.




*(a)* LoRA rank 8, Population 48




*(b)* LoRA rank 16, Population 96



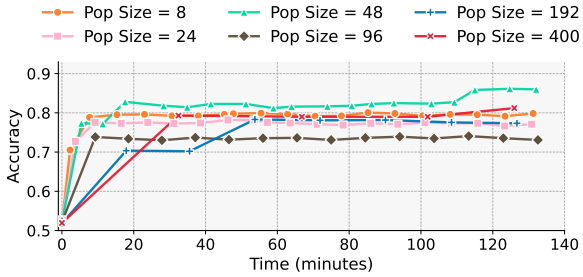

*(c)* LoRA rank 32, Population 96

*Figure 15.* Validation accuracy on **GSM8K** with **Qwen2.5-7B** while varying $\alpha$. LoRA rank and population size are fixed to the optimal choices from Figure 9. Batch size 100.

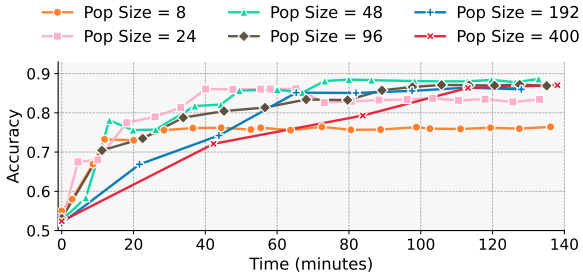
Figure 15 shows the full training dynamics of ESSA on GSM8K with Qwen2.5-7B when varying the fraction $\alpha$ of trainable singular values.

### C.2.2. Qwen2.5-Math-7B on GSM8K

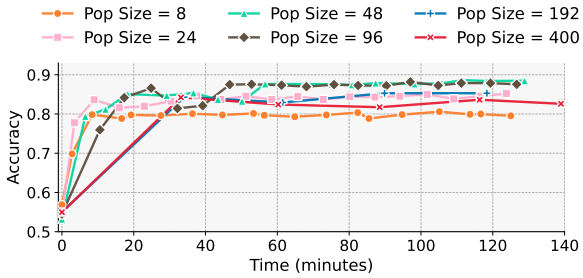
Appendix Figures 16, 17 and 18 provide the complete training-dynamic study of ESSA on Qwen2.5-Math-7B for GSM8K. These graphs complement the main text sensitivity analysis by showing how validation accuracy evolves over time as we vary the three key ESSA hyperparameters (Figure 2).



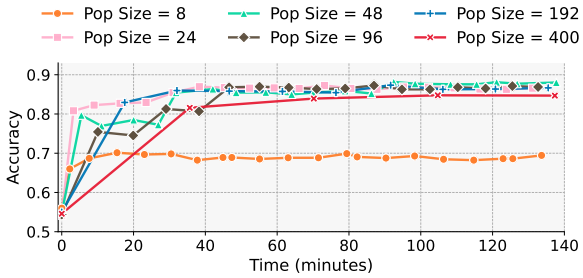

*(a)* LoRA rank 2



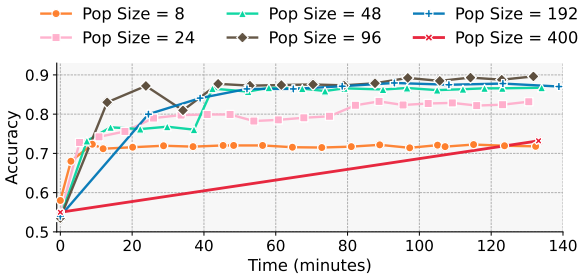

*(b)* LoRA rank 4




*(c)* LoRA rank 8



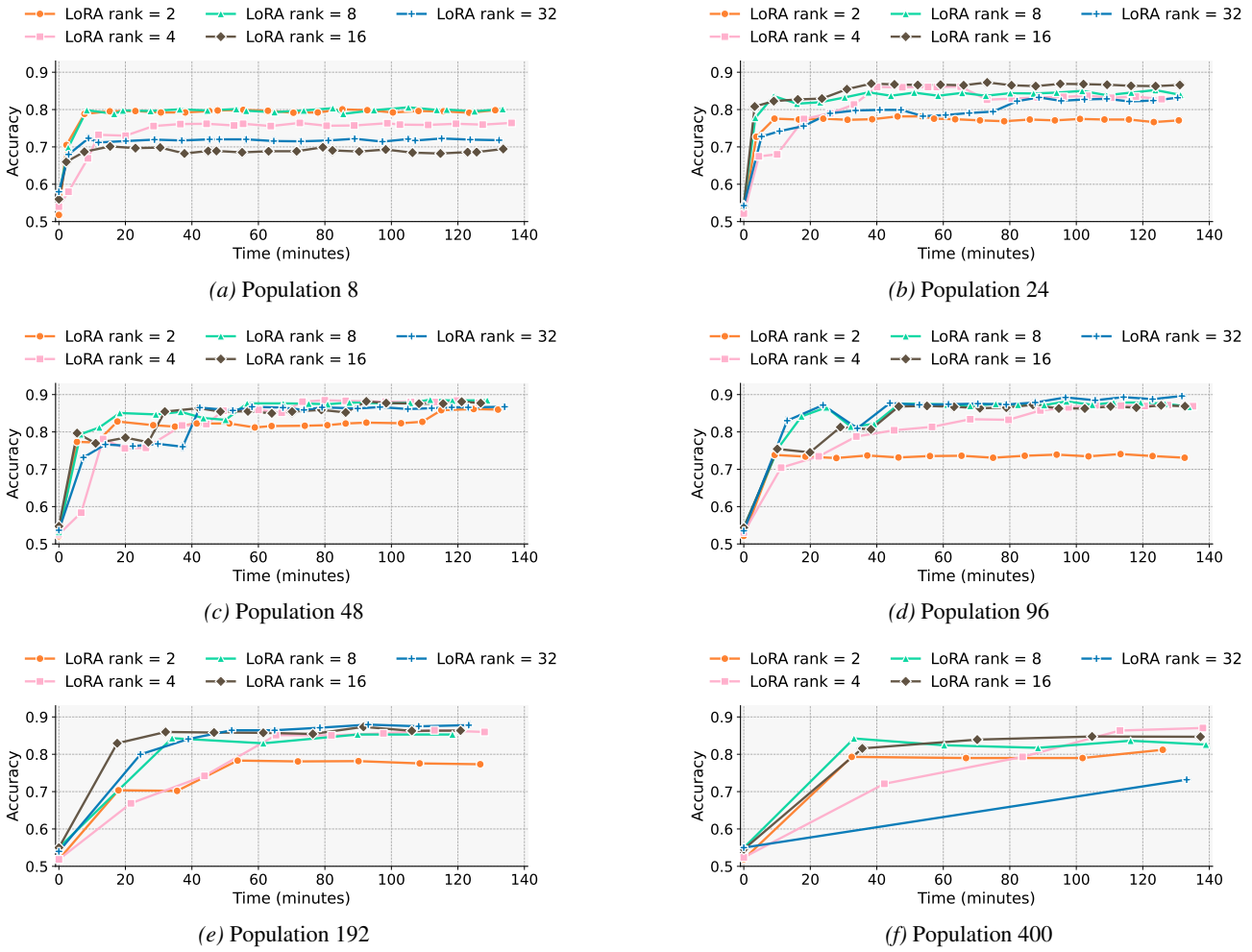

*(d)* LoRA rank 16




*(e)* LoRA rank 32

*Figure 16.* Validation accuracy over time on **GSM8K** with **Qwen2.5-Math-7B** when varying the ESSA population size (8, 24, 48, 96, 192, 400) at fixed LoRA ranks (2, 4, 8, 16, 32). Batch size 100.

Figure 16 shows the full training dynamics of ESSA on GSM8K with Qwen2.5-Math-7B when varying the ESSA population size.




*(a)* Population 8

*(b)* Population 24

*(c)* Population 48

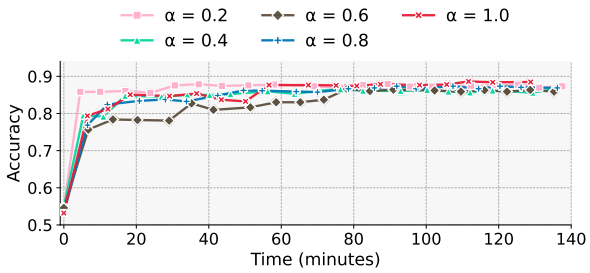
*(d)* Population 96

*(e)* Population 192

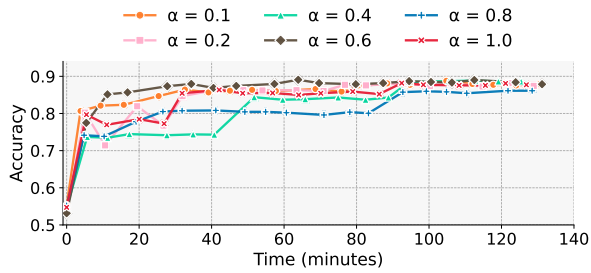
*(f)* Population 400

*Figure 17.* Validation accuracy over time on **GSM8K** with **Qwen2.5-Math-7B** when varying the LoRA rank (2, 4, 8, 16, 32) at fixed population sizes (8, 24, 48, 96, 192, 400). Batch size 100.

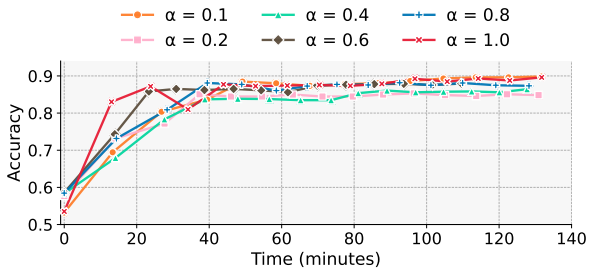
Figure 17 shows the full training dynamics of ESSA on GSM8K with Qwen2.5-Math-7B when varying the the LoRA rank.




*(a)* LoRA rank 8, Population 48




*(b)* LoRA rank 16, Population 48



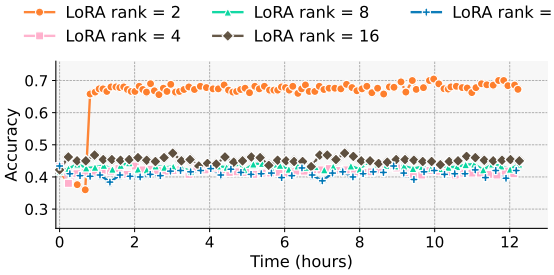

*(c)* LoRA rank 32, Population 96

*Figure 18.* Validation accuracy on **GSM8K** with **Qwen2.5-Math-7B** while varying $\alpha$. LoRA rank and population size are fixed to the optimal choices from Figure 2. Batch size 100.

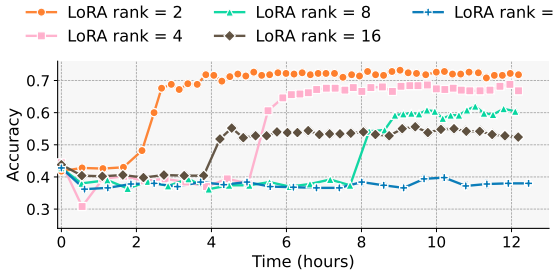
Figure 18 shows the full training dynamics of ESSA on GSM8K with Qwen2.5-Math-7B when varying the fraction $\alpha$ of trainable singular values.

### C.2.3. Qwen2.5-7B on PRM800K

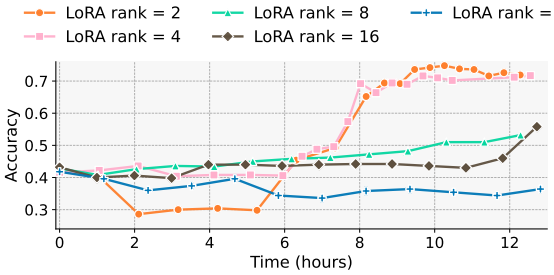
Appendix Figures 19, 20 and 21 provide the complete training-dynamic study of ESSA on Qwen2.5-7B for PRM800K (evaluated on MATH500). This analysis shows how validation accuracy evolves over time as we vary the three key ESSA hyperparameters (Figure 10)



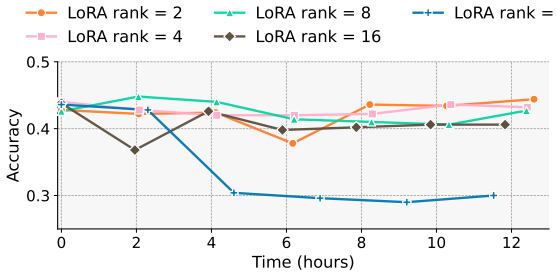

*(a)* Population 8



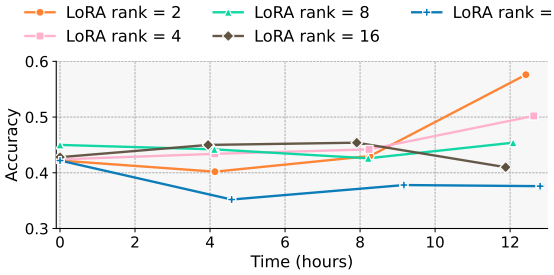

*(b)* Population 24




*(c)* Population 48



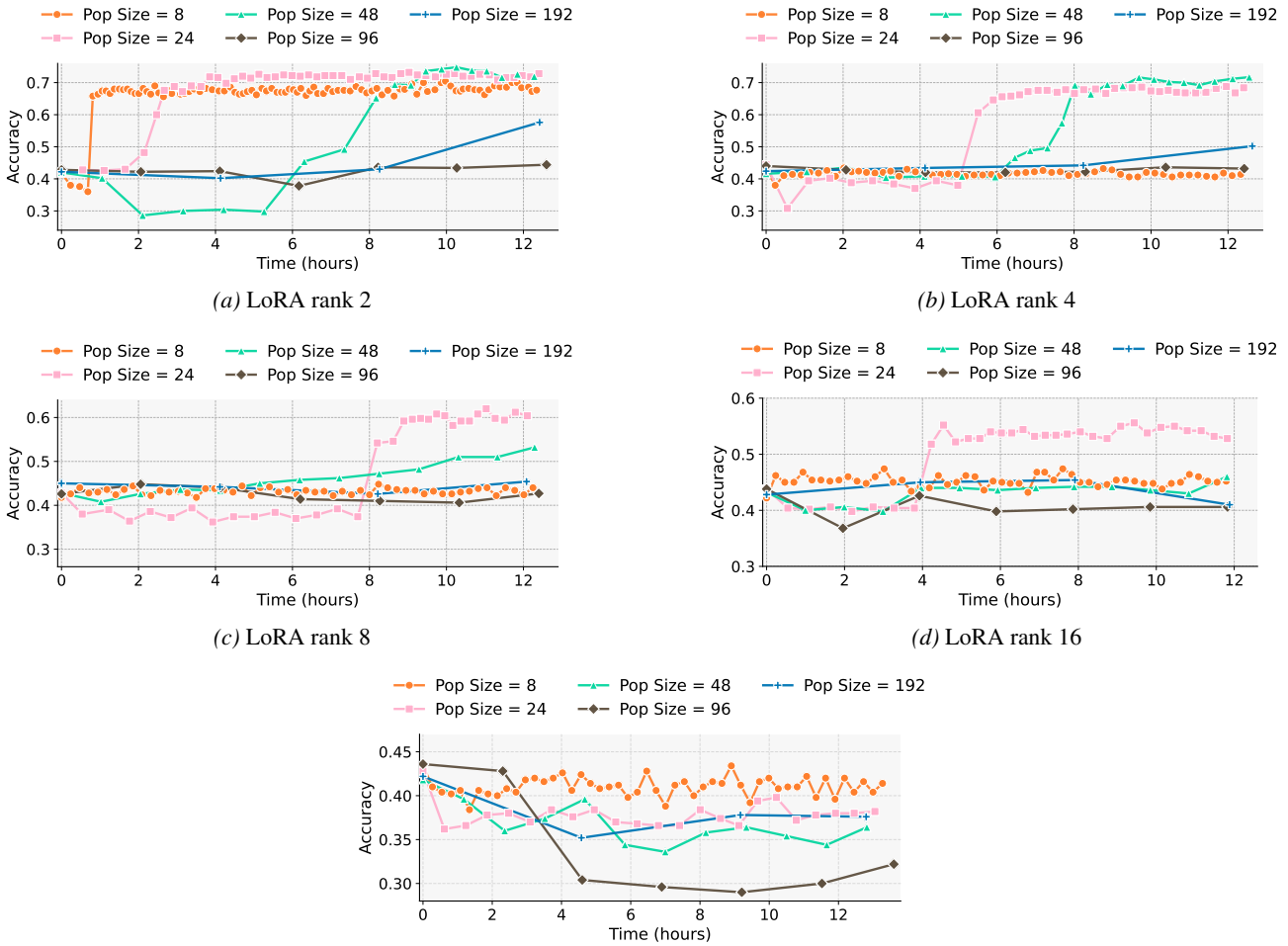

*(d)* Population 96




*(e)* Population 192

*Figure 19.* Validation accuracy over time on **PRM800K** with **Qwen2.5-7B** when varying the ESSA population size (8, 24, 48, 96, 192) at fixed LoRA ranks (2, 4, 8, 16, 32). Batch size 300.

Figure 19 shows the full training dynamics of ESSA on PRM800K (evaluated on MATH500) with Qwen2.5-7B when varying the ESSA population size.




*(a)* LoRA rank 2

*(b)* LoRA rank 4

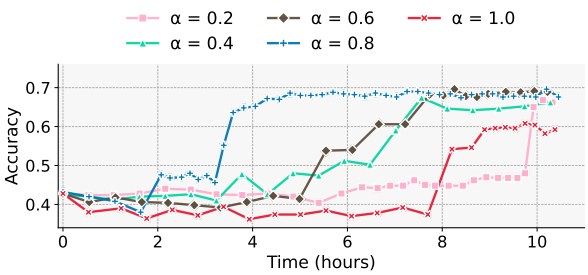
*(c)* LoRA rank 8

*(d)* LoRA rank 16

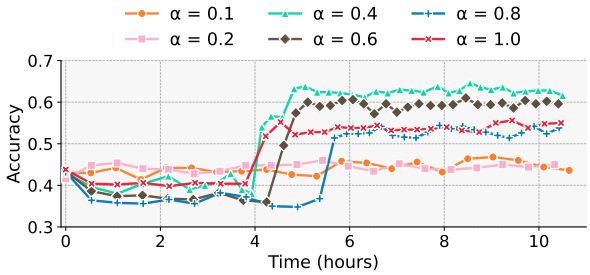
*(e)* LoRA rank 32

*Figure 20.* Validation accuracy over time on **PRM800K** with **Qwen2.5-7B** when varying the LoRA rank (2, 4, 8, 16, 32) at fixed population sizes (8, 24, 48, 96, 192). Batch size 300.

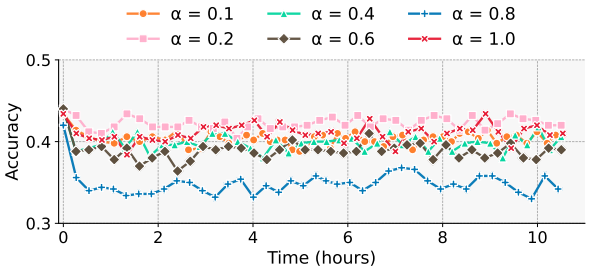
Figure 20 shows the full training dynamics of ESSA on PRM800K (evaluated on MATH500) with Qwen2.5-7B when varying the LoRA rank.




*(a)* LoRA rank 8, Population 24




*(b)* LoRA rank 16, Population 24



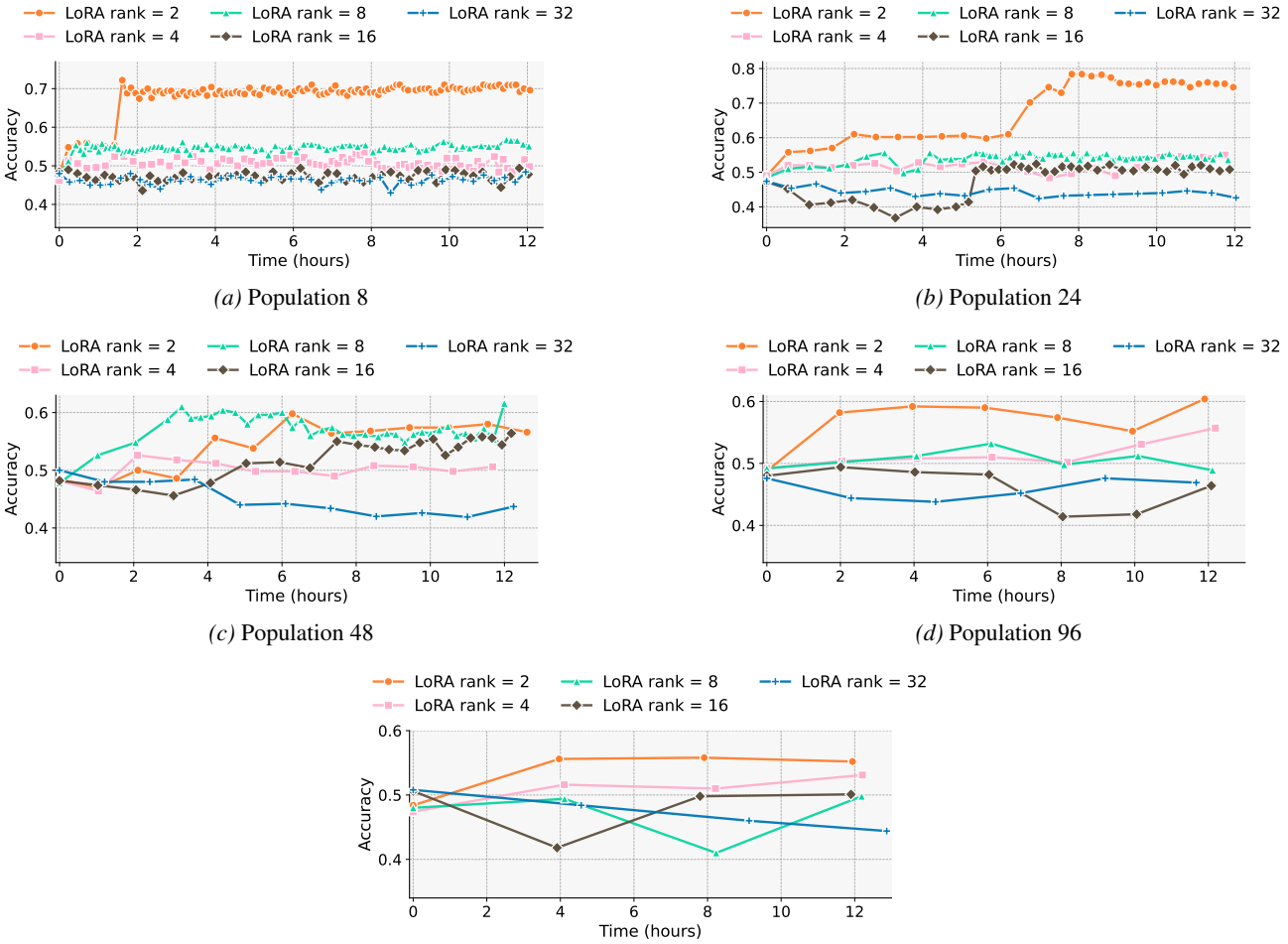

*(c)* LoRA rank 32, Population 8

*Figure 21.* Validation accuracy over time on **PRM800K** with **Qwen2.5-7B** while varying $\alpha$. LoRA rank and population size are fixed to the optimal choices from Figure 10. Batch size 300.

Figure 21 shows the full training dynamics of ESSA on PRM800K (evaluated on MATH500) with Qwen2.5-7B when varying the fraction $\alpha$ of trainable singular values.

### C.2.4. Qwen2.5-Math-7B on PRM800K

Appendix Figures 22, 23 and 24 provide the complete training-dynamic study of ESSA on Qwen2.5-Math-7B for PRM800K (evaluated on MATH500). These graphs complement the main text sensitivity analysis by showing how validation accuracy evolves over time as we vary the three key ESSA hyperparameters (Figure 11).




*(a)* Population 8

*(b)* Population 24

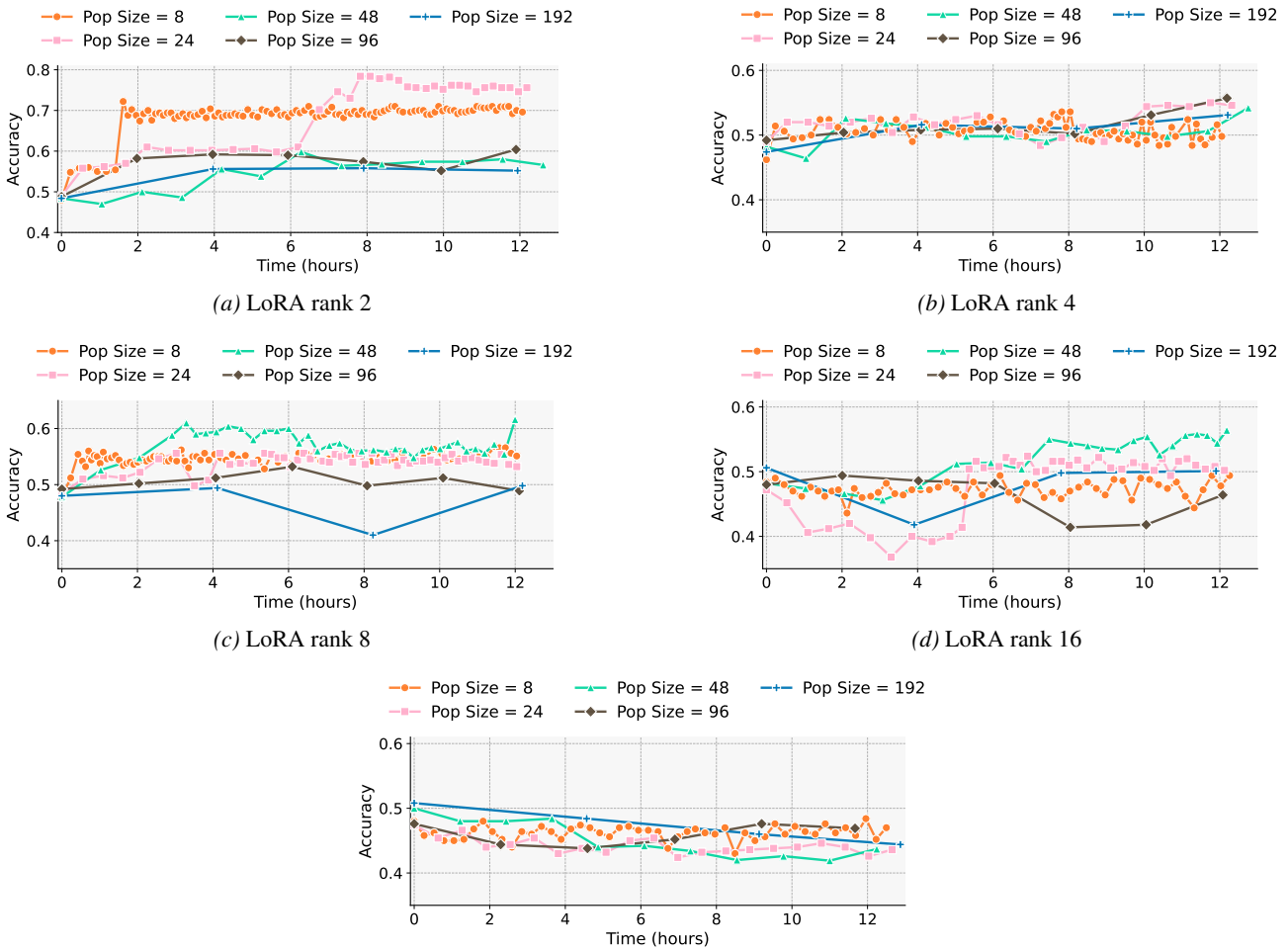
*(c)* Population 48

*(d)* Population 96

*(e)* Population 192

*Figure 22.* Validation accuracy over time on **PRM800K** with **Qwen2.5-Math-7B** when varying the ESSA population size (8, 24, 48, 96, 192) at fixed LoRA ranks (2, 4, 8, 16, 32). Batch size 300.

Figure 22 shows the full training dynamics of ESSA on PRM800K (evaluated on MATH500) with Qwen2.5-Math-7B when varying the ESSA population size.




*(a)* LoRA rank 2

*(b)* LoRA rank 4

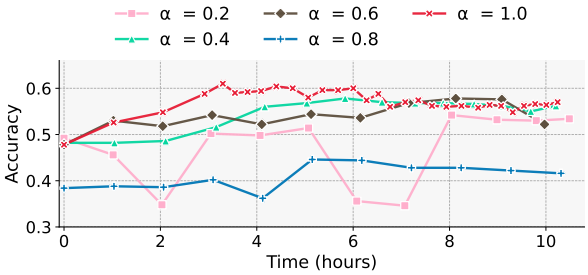
*(c)* LoRA rank 8

*(d)* LoRA rank 16

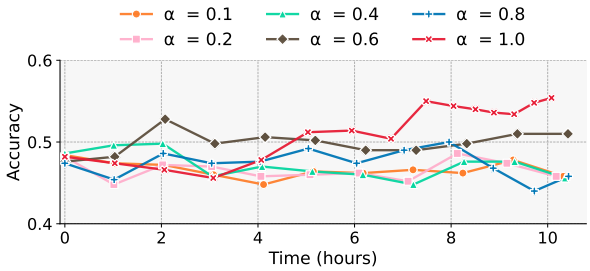
*(e)* LoRA rank 32

*Figure 23.* Validation accuracy over time on **PRM800K** with **Qwen2.5-Math-7B** when varying the LoRA rank (2, 4, 8, 16, 32) at fixed population sizes (8, 24, 48, 96, 192). Batch size 300.

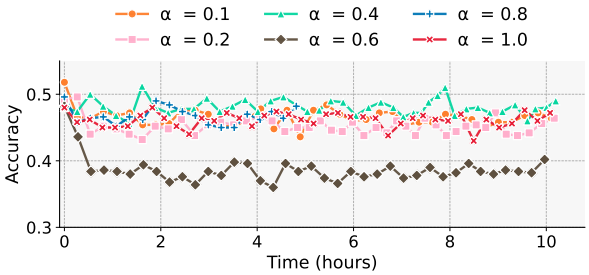
Figure 23 shows the full training dynamics of ESSA on PRM800K (evaluated on MATH500) with Qwen2.5-Math-7B when varying the LoRA rank.




*(a)* LoRA rank 8, Population 48




*(b)* LoRA rank 16, Population 48



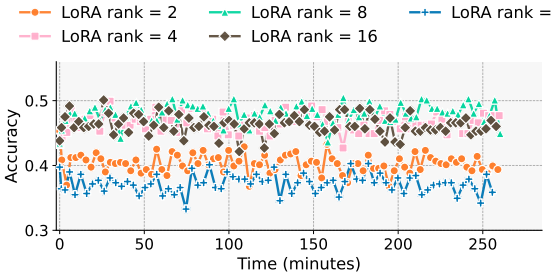

*(c)* LoRA rank 32, Population 8

*Figure 24.* Validation accuracy over time on **PRM800K** with **Qwen2.5-Math-7B** when varying $\alpha$. LoRA rank and population size are fixed to the optimal choices from Figure 11. Batch size 300.

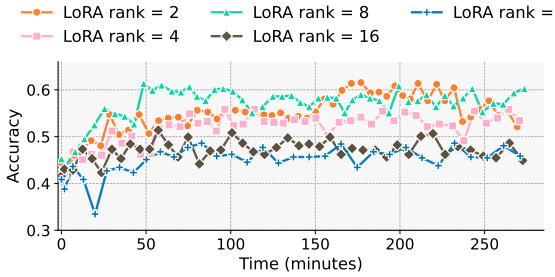
Figure 24 shows the full training dynamics of ESSA on PRM800K (evaluated on MATH500) with Qwen2.5-Math-7B when varying the fraction $\alpha$ of trainable singular values.

### C.2.5. LLaMA3.1-8B on IFEval

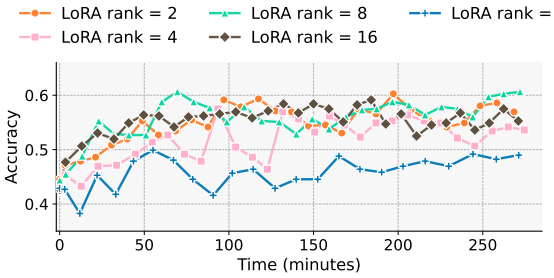
Appendix Figures 25 and 26 provide the complete training-dynamic study of ESSA on LLaMA3.1-8B for IFEval. Two graphs complement the main text sensitivity analysis by showing how validation accuracy evolves over time as we vary the two key ESSA hyperparameters (Figure 12)



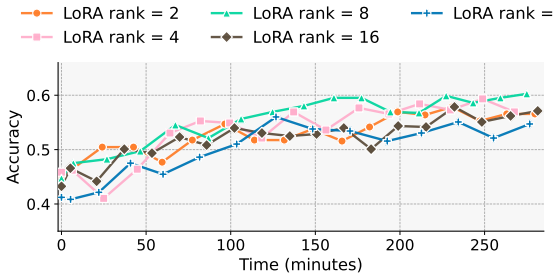

*(a)* Population 8



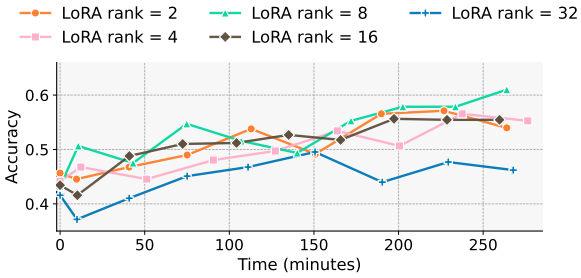

*(b)* Population 24




*(c)* Population 48



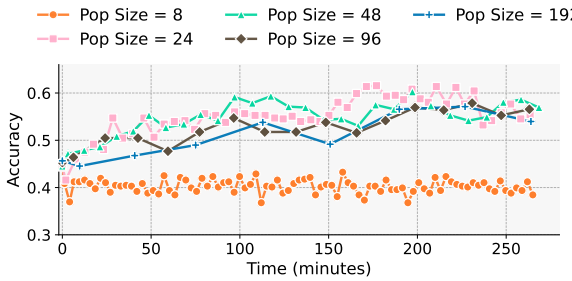

*(d)* Population 96



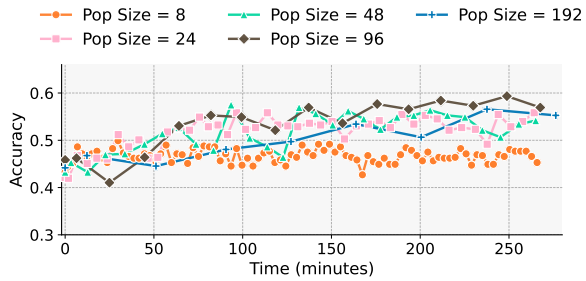

*(e)* Population 192

*Figure 25.* Validation accuracy over time on **IFEval** with **LLaMA3.1-8B** when varying the ESSA population size (8, 24, 48, 96, 192) at fixed LoRA ranks (2, 4, 8, 16, 32). Batch size 500, $\alpha = 1.0$.

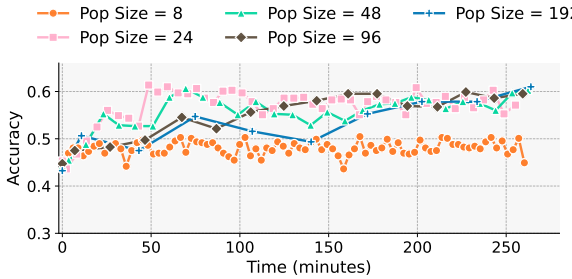
Figure 25 shows the full training dynamics of ESSA on IFEval with LLaMA3.1-8B when varying the ESSA population size.



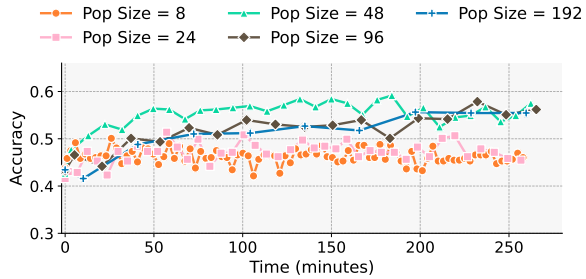

*(a)* LoRA rank 2



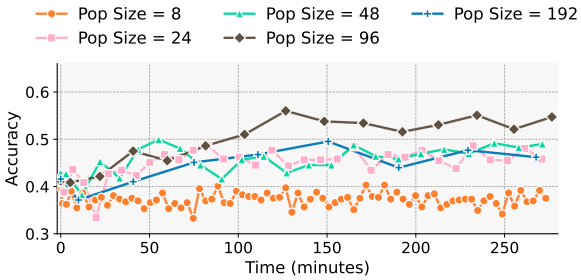

*(b)* LoRA rank 4




*(c)* LoRA rank 8




*(d)* LoRA rank 16




*(e)* LoRA rank 32

*Figure 26.* Validation accuracy over time on **IFEval** with **LLaMA3.1-8B** when varying the LoRA rank (2, 4, 8, 16, 32) at fixed population sizes (8, 24, 48, 96, 192). Batch size 500, $\alpha = 1.0$.

Figure 26 shows the full training dynamics of ESSA on IFEval with LLaMA3.1-8B when varying the LoRA rank.

## D. Precision Analysis

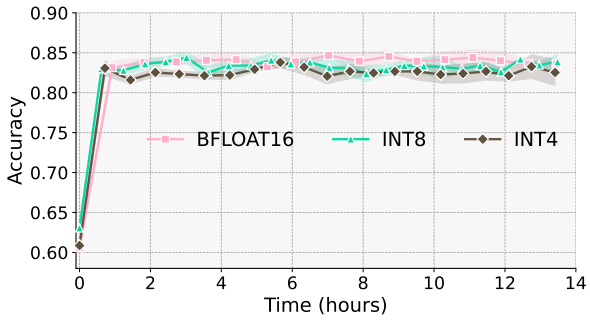



*Figure 27.* MATH500 accuracy over training time for Qwen2.5-32B on PRM800K under different weight precisions (BFLOAT16, INT8, INT4). Settings: LoRA rank 8, pop. 64, batch size 256, $\alpha = 1.0$.

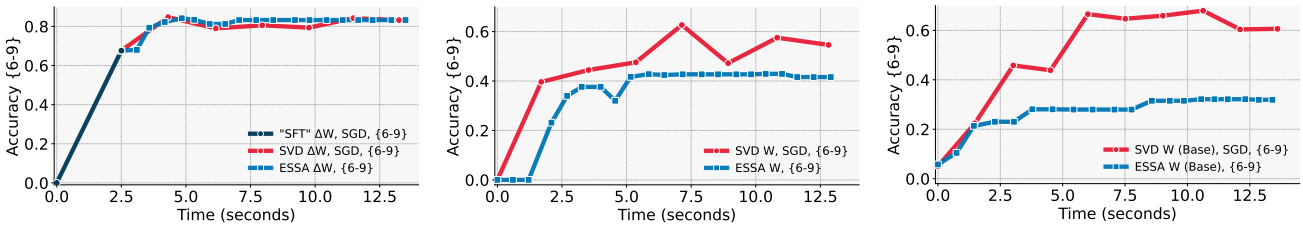
Figure 27 shows the full training curves of ESSA when running Qwen2.5-32B on PRM800K (evaluated on MATH500) with LoRA rank 8, population size 64, $\alpha = 1.0$, and per-candidate batch size 256, for three different numerical precisions: BFLOAT16, INT8, and INT4. Shaded areas indicate one standard deviation across runs.

Across all precisions, accuracy rises quickly within the first hour and then remains stable. Reducing precision from BFLOAT16 to INT8 and even to INT4 produces only a slight reduction in final accuracy, consistent with the summary in Table 3 of the main text. These results confirm that ESSA can safely exploit aggressive quantization to cut memory usage and computation cost while maintaining near-identical performance.

## E. Toy Example on MNIST

A natural concern arises when considering a singular-value-only parameterization. If we restrict the problem to simple linear or logistic regression, scaling the rows of a fixed low-rank update may seem too limited to support meaningful adaptation. In this setting only the singular magnitudes change while all directions remain fixed, which can give the impression of insufficient flexibility.

Motivated by this concern, we emphasize that ESSA's effectiveness in LLMs relies on the structured representations already present in the pretrained backbone and inherited through the downstream SFT initialization. To make this distinction concrete, we designed an MNIST experiment that replicates the LLM training pipeline as closely as possible within a minimal linear model.




*(a)* $W = W' + \Delta W$ *(b)* $W = W'$ *(c)* $W \sim \mathcal{N}(0, \frac{2}{n})$

*Figure 28.* Validation accuracy over time on **MNIST** (on the test set of digits 6-9) with a single matrix $\mathbf{W} \in \mathbb{R}^{n \times m}$, $n = 784$, $m = 10$. Panels (a)-(c) correspond to $W = W' + \Delta W$, $W = W'$ and $W \sim \mathcal{N}(0, \frac{2}{n})$ setups respectively, where $W'$ is pretrained base weights. Train size 400. ESSA (blue): solution length 10, population size 24, batch size 64. SGD (red): lr 0.1, batch size 64.

Toy example on MNIST proceeds as follows:

1. At the pretrain stage (analog of LLM pretraining) we train a logistic regression model on MNIST digits 0-5 (train size 5000) using SGD optimizer. We obtain a "base model" $W = W'$, where $W'$ is the single matrix of size 784 by 10, that knows something about the data distribution, similar to LLM pretraining.
2. At the "SFT" stage (analogue of LoRA SFT) we take 400 samples of digits 6-9, initialize a matrix update $\Delta W$ with Kaiming init, and train only $\Delta W$ using SGD: $W = W' + \Delta W$, where $\Delta W$ is the single matrix of size 784 by 10.

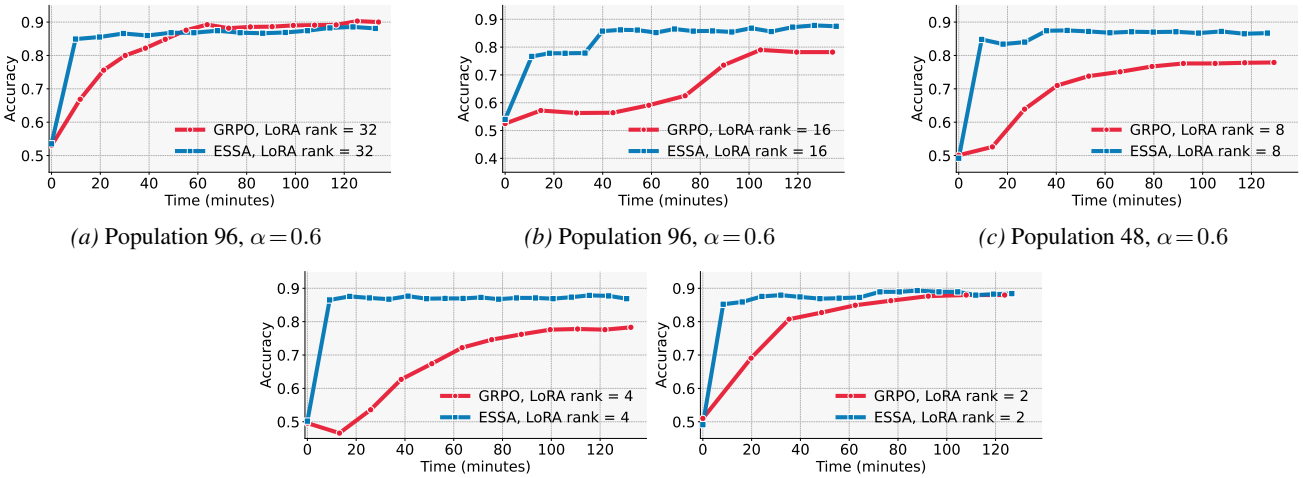
3. At the last stage we decomposed $\Delta W = BSA$ and train only the singular values $S$ using ES or SGD optimizer. ESSA optimize singular values directly using train accuracy, while the SGD baseline optimized them through the standard cross-entropy loss.

In Figure 28a, ESSA follows the behavior of SGD on the MNIST task with high fidelity. This demonstrates that once the singular vectors encodes useful directions, modulating only the associated singular amplitudes is already sufficient to achieve effective adaptation. In Figure 28b, we remove the additive update entirely. We decompose the pretrained weights $W = W' = B'S'A'$, and train only the singular values $S'$ using ESSA and SGD. Even in this more constrained setup, ESSA achieved performance comparable to SGD. The pretrained matrix contained enough structure that rebalancing its singular directions remained a viable strategy for improving accuracy. Finally, in Figure 28c, we examine the case without pretraining or SFT, where the model is initialized with and the singular vectors carry no semantic information. In this setup ESSA improves only slowly, while SGD makes substantially faster progress. Without a structured subspace, changing only singular values is insufficient to emulate full learning dynamics.

These experiments demonstrate that the singular-value parameterization is effective precisely in the settings where LLMs operate: pretrained models with downstream SFT produce meaningful low-rank directions, and ESSA leverages this structure by adjusting their relative scaling. This behavior is consistent with the results observed in our large-scale LLM experiments.

# F. Comparison with GRPO

## F.1. School Math

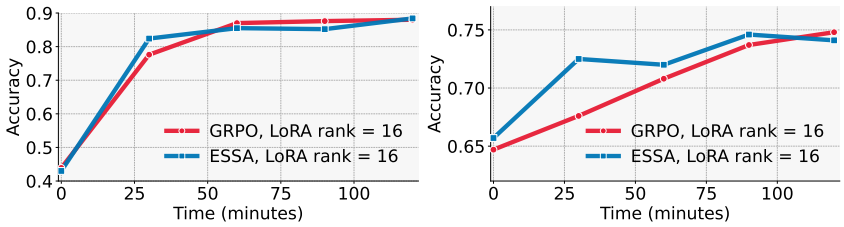



*(a)* Population 96, $\alpha = 0.6$ *(b)* Population 96, $\alpha = 0.6$ *(c)* Population 48, $\alpha = 0.6$

*(d)* Population 96, $\alpha = 1.0$ *(e)* Population 96, $\alpha = 1.0$

*Figure 29.* Validation accuracy over time on **GSM8K** with **Qwen2.5-7B**. Panels (a)-(e) correspond to LoRA ranks 32, 16, 8, 4, and 2, respectively. ESSA (blue): batch size 100. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64.

Appendix Figure 29 presents the results of Qwen2.5-7B on GSM8K in addition to Qwen2.5-Math-7B (Figure 4).




*(a)* Population 96, $\alpha = 1.0$ *(b)* Population 96, $\alpha = 1.0$

*Figure 30.* Validation accuracy over time on **GSM8K** with **Qwen2.5-7B** (a) and **Qwen2.5-Math-7B** (b) without SFT warm-start. ESSA (blue): train size 100. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64.

Appendix Figure 30 presents the results of Qwen2.5-7B and Qwen2.5-Math-7B on GSM8K without SFT warm-start in addition to School Math experiments (Section 4.4.1). LoRA matrix $A$ is initialized with Kaiming initialization, $B$ is zero.

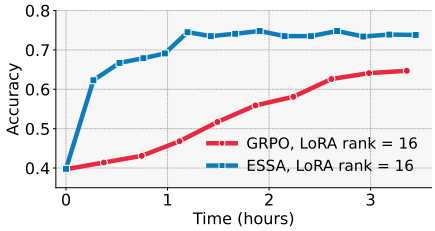
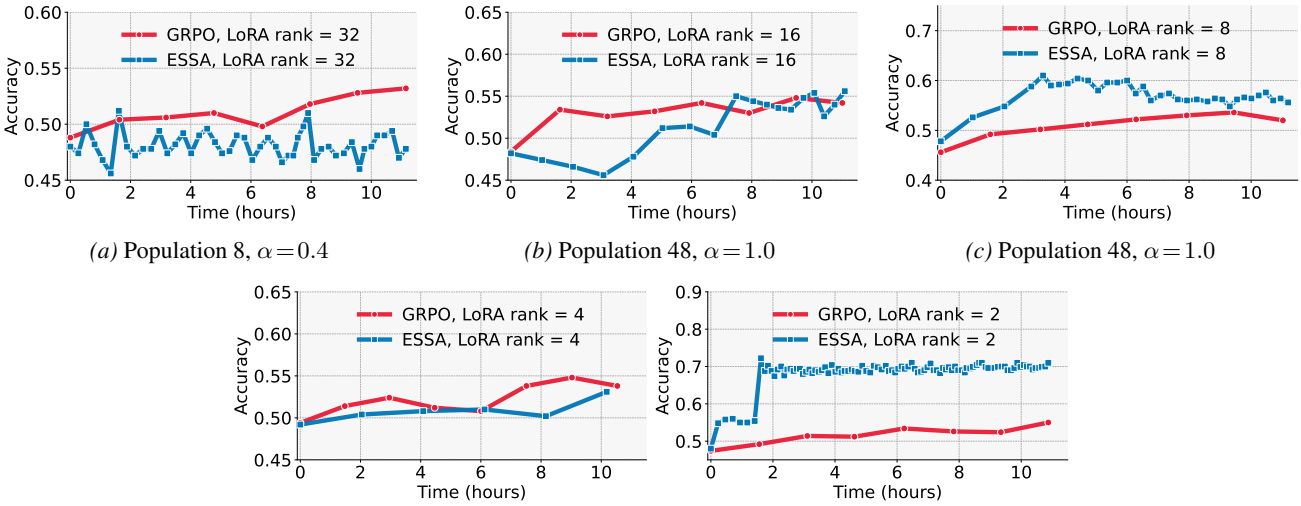



*Figure 31.* Validation accuracy over time on **GSM8K** with **Qwen2.5-3B** on one GPU. ESSA (blue): population size 48, $\alpha = 1.0$, batch size 100. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64.

Appendix Figure 31 presents the results of Qwen2.5-3B on one GPU in addition to School Math experiments (Section 4.2).

### F.2. Beyond School Math

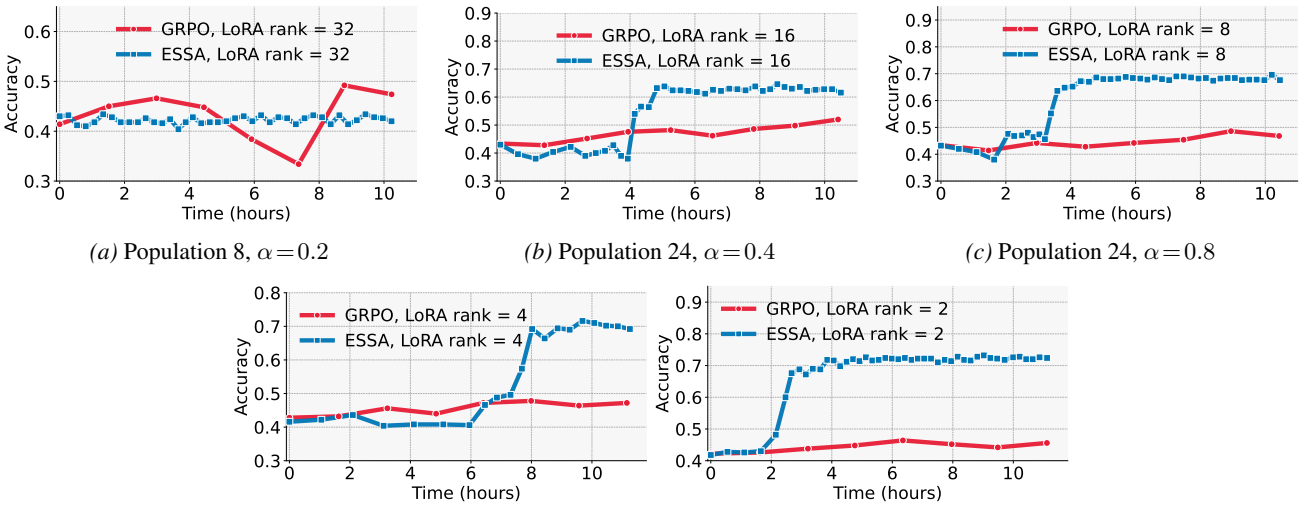



*(a)* Population 8, $\alpha = 0.4$ *(b)* Population 48, $\alpha = 1.0$ *(c)* Population 48, $\alpha = 1.0$

*(d)* Population 96, $\alpha = 1.0$ *(e)* Population 8, $\alpha = 1.0$

*Figure 32.* Validation accuracy over time on **PRM800K** with **Qwen2.5-Math-7B**. Panels (a)-(e) correspond to LoRA ranks 32, 16, 8, 4, and 2, respectively. ESSA (blue): batch size 300. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64.

Appendix Figures 32 and 33 present the full results of our beyond school math experiments on PRM800K (validation on MATH500).




*(a)* Population 8, $\alpha = 0.2$ *(b)* Population 24, $\alpha = 0.4$ *(c)* Population 24, $\alpha = 0.8$

*(d)* Population 48, $\alpha = 1.0$ *(e)* Population 24, $\alpha = 1.0$

*Figure 33.* Validation accuracy over time on **PRM800K** with **Qwen2.5-7B**. Panels (a)-(e) correspond to LoRA ranks 32, 16, 8, 4, and 2, respectively. ESSA (blue): batch size 300. GRPO (red): lr $1 \times 10^{-5}$, global batch 512, mini batch 64.

We evaluate both Qwen2.5-Math-7B and Qwen2.5-7B, sweeping LoRA ranks and comparing ESSA with GRPO. Across both model variants on GSM8K (Section 4.4.1, Figure 29) and PRM800K (Section 4.4.2, Section F.2) ESSA climbs to high accuracy much more rapidly than GRPO and often reaches a higher plateau, particularly when the LoRA rank is small or moderate. In experiments without SFT (Figure 30) runs starting from a random LoRA achieve lower absolute accuracy. The relative performance between ESSA and GRPO remains consistent. Experiment with Qwen2.5-3B (Figure 31) on one GPU confirms that ESSA's benefits are not restricted to multi-GPU setups, but persist even when only one device is available.

## G. Comparison with other Evolutionary Methods

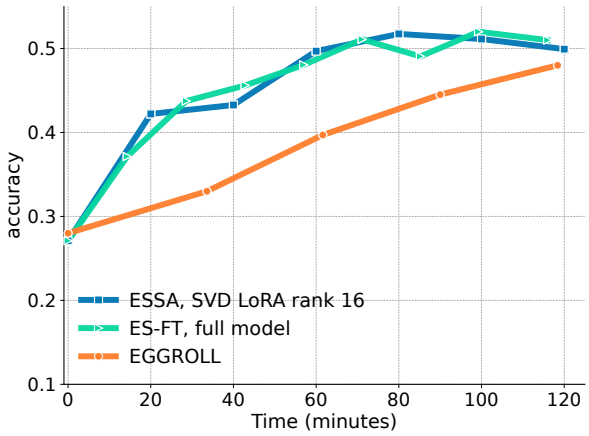
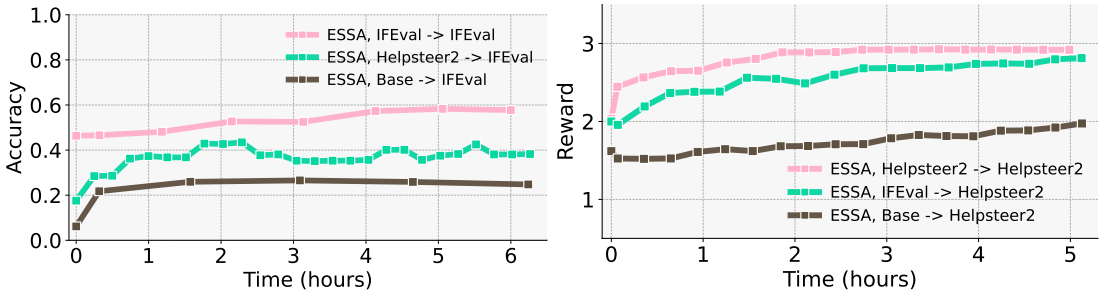



*Figure 34.* Validation accuracy over time on **Countdown** with **Qwen2.5-7B** on 8 GPUs. ESSA (blue): population 48, $\alpha = 1.0$, batch size 200. ES-FT (green): SIGMA $= 1\times10^{-3}$, ALPHA $= 1\times10^{-4}$, population 48, batch size 200. EGGROLL (orange): SIGMA $= 1\times10^{-3}$, rank 1, batch size 200.

Appendix Figure 34 compares the performance of three evolutionary methods – ESSA, ES-FT, and EGGROLL – in optimizing the Qwen2.5-7B model on the Countdown task using 8 GPUs. This experiment demonstrates that ESSA significantly outperforms EGGROLL and matches the convergence of the full-model fine-tuning approach ES-FT.

## H. OOD Drift




*(a)* LoRA rank 8, population 48, $\alpha = 1.0$ *(b)* LoRA rank 8, population 36, $\alpha = 1.0$

*Figure 35.* Validation metrics over time on **IFEval** (a) and **HelpSteer2** (b) with **Llama3.1-8B**. (a) IFEval → IFEval (pink), HelpSteer2 → IFEval (green), base model → IFEval (brown) experiments, batch size 500. (b): HelpSteer2 → HelpSteer2 (pink), IFEval → HelpSteer2 (green), base model → HelpSteer2 (brown) experiments, batch size 100.

To assess the robustness of ESSA under distribution shift, we conduct an additional set of experiments that instantiate exactly the cross-domain setting proposed in the review. We select two datasets with markedly different reward structures: HelpSteer2 and IFEval. HelpSteer2 encourages general-purpose response quality, while IFEval enforces rigid constraint satisfaction. Moving between them therefore offers a direct test of how well ESSA adapts when the SFT warm-start and the target alignment domain differ. All HelpSteer2 experiments use the same reward model as in the main paper (RLHFlow/ArmoRM-Llama3-8B-v0.1). All IFEval experiments use if-eval-like dataset as a train dataset.

For each target dataset B we evaluate three configurations. The first performs SFT on dataset A and ESSA on B (A → B). The second performs both SFT and ESSA on B (B → B). The third applies ESSA directly to B without any SFT warm-start (Base model → B), initialized with default LoRA parameters (Kaiming-uniform for weight A and zeros for weight B).

Across both directions, ESSA reliably improves performance on the target dataset, even when the SFT checkpoint originates from a mismatched domain. In the IFEval to HelpSteer2 condition the gap between cross-domain and in-domain SFT is modest, indicating that the structured initialization learned from if-eval-like dataset already provides useful axes for ESSA to exploit when optimizing for HelpSteer2. In the opposite direction the gap is larger, which is expected given that HelpSteer2

does not teach the model the strict formatting and constraint-following behavior required by IFEval. Nevertheless ESSA exhibits stable monotonic improvement and substantially surpasses the Base → IFEval configuration that starts from random LoRA parameters. In both directions, SFT warm-starts clearly outperform training from scratch, confirming that even an out-of-distribution SFT checkpoint supplies meaningful structure for ESSA's evolutionary search.

These results demonstrate that ESSA maintains its effectiveness under distribution shift and that the method can leverage any reasonable SFT initialization, even when it originates from a domain with incompatible reward structure.

# I. Implementation Details

## I.1. Datasets

| Dataset | SFT Train Size | ESSA/GRPO Train Size | Validation |
|---|---|---|---|
| GSM8K | 1495 | 5978 | 1319 |
| PRM800K | 3600 | 7900 | 500 |
| IFEval | 1700 | 3414 | 541 |
| HelpSteer2 | 19000 | 1324 | 1024 |
| Countdown | – | 1000 | 1024 |

*Table 6.* Dataset sizes across all experiments. For each dataset, the "SFT Train Size" and "ESSA/GRPO Train Size" are disjoint partitions of the dataset's training split.

### System Prompts

- **Math-Reasoning:** `Please reason step by step and place your final answer in \boxed{}.`
- **Countdown:** `You are a helpful assistant.`
- **Instruction-Following:** `You are a helpful, honest, and concise assistant.`

## I.2. SFT Hyperparameters

| Model and Dataset | SFT Batch Size | Learning Rate | Epochs | SFT GPU Hours |
|---|---|---|---|---|
| Qwen2.5-7B (GSM8K) | 16 | $1 \times 10^{-4}$ | 1 | 0.18 |
| Qwen2.5-Math-7B (GSM8K) | 16 | $1 \times 10^{-4}$ | 1 | 0.16 |
| Qwen2.5-7B (PRM800K) | 16 | $5 \times 10^{-4}$ | 1 | 0.43 |
| Qwen2.5-Math-7B (PRM800K) | 16 | $5 \times 10^{-4}$ | 1 | 0.40 |
| Qwen2.5-32B (PRM800K) | 128 | $1 \times 10^{-6}$ | 3 | 9.80 |
| Qwen2.5-72B (PRM800K) | 128 | $1 \times 10^{-6}$ | 3 | 51.97 |
| Llama3.1-8B (IF-Eval-like) | 16 | $5 \times 10^{-4}$ | 1 | 0.50 |
| Llama3.1-8B (HelpSteer2) | 128 | $1 \times 10^{-6}$ | 3 | 3.74 |

*Table 7.* Summary of hyperparameters used for SFT across all experiments.

## I.3. ESSA Hyperparameters

For ESSA, we performed a hyperparameter grid search covering various values of LoRA rank, population size, fraction of singular values $\alpha$, batch size, and precision (Table 8). For each experiment, the best ESSA hyperparameters were selected based on validation performance (Table 9). We use up to 128 NVIDIA H100 80GB GPUs per experiment.

| Parameter | Values Tested | Default |
|---|---|---|
| LoRA Rank | 2, 4, 8, 16, 32 | 8 |
| Population Size | 8, 24, 36, 48, 64, 96, 192, 400 | 96 |
| Fraction of Singular Values ($\alpha$) | 0.1, 0.2, 0.4, 0.6, 0.8, 1.0 | 1.0 |
| Batch Size | 100, 256, 300, 500 | 100 |
| Precision | INT4, INT8, BFLOAT16 | BFLOAT16 |
| CMA-ES initial variance | 0.32 | 0.32 |

*Table 8.* ESSA hyperparameter grid and default values for ablation studies.

| Experiment | LoRA Rank | Population Size | Batch Size | Fraction $\alpha$ |
|---|---|---|---|---|
| GSM8K (Qwen2.5-Math-7B) | 32 | 96 | 100 | 0.1 |
| GSM8K (Qwen2.5-7B) | 2 | 96 | 100 | 1.0 |
| PRM800K (Qwen2.5-Math-7B) | 2 | 24 | 300 | 1.0 |
| PRM800K (Qwen2.5-7B) | 2 | 48 | 300 | 1.0 |
| PRM800K (Qwen2.5-32B) | 8 | 48 | 256 | 1.0 |
| PRM800K (Qwen2.5-72B, INT4) | 4 | 64 | 256 | 1.0 |
| IFEval (LLaMA3.1-8B) | 8 | 24 | 500 | 1.0 |
| HelpSteer (LLaMA3.1-8B) | 8 | 36 | 100 | 1.0 |
| Countdown (Qwen2.5-7B) | 16 | 48 | 200 | 1.0 |

*Table 9.* Best ESSA hyperparameters for each experiment.