Balanced Aggregation: Understanding and Fixing Aggregation Bias in GRPO

Anonymous ACL submission

Abstract

Reinforcement learning with verifiable rewards (RLVR) has become a central paradigm for improving reasoning and code generation in large language models, and GRPO-style training is widely adopted for its simplicity and effectiveness. However, an important design choice remains underexplored: how token-level policy gradient terms are aggregated within each sampled group. Standard GRPO uses sequence aggregation, while recent work has advocated token aggregation as a better alternative. We show that these two rules induce different optimization biases: token aggregation introduces sign-length coupling, while sequence aggregation implicitly downweights longer responses through sequence-level equal weighting. To address this tension, we propose **Balanced Aggregation (BA)**, a simple drop-in replacement that computes token-level means separately within the positive and negative subsets and then combines them with sequence-count-based weights. Experiments with Qwen2.5-Math-7B and Qwen3-1.7B on DAPO-17k and Polaris, evaluated on six reasoning and coding benchmarks, show that BA consistently improves training stability and final performance over standard token and sequence aggregation. Our analysis further shows that the relative effectiveness of token and sequence aggregation is largely governed by response-length variation and the positive-negative length gap, highlighting aggregation as a critical design dimension in GRPO-style RLVR.

1 Introduction

Reinforcement learning with verifiable rewards (RLVR) has become a central paradigm for improving the reasoning and code generation abilities of large language models (LLMs). By replacing learned reward models with programmatically verifiable signals such as exact-match correctness or unit-test pass rate, RLVR provides a simple and scalable way to optimize models on tasks with objective outcomes.

Among recent RLVR methods, GRPO-style training is particularly attractive in practice due to its simplicity and effectiveness. For each prompt, the policy samples multiple responses, assigns rewards based on verifiable outcomes, and computes normalized group-wise advantages to optimize a PPO-style objective. This design has been widely adopted in reasoning and coding settings because it avoids training a separate critic while still providing useful relative learning signals within each sampled group.

Despite the growing adoption of GRPO-style RLVR, an important design choice remains underexplored: *how token-level policy gradient terms are aggregated within each sampled group*. In standard GRPO, the default choice is *sequence aggregation*, which first averages over tokens within each response and then averages across responses. Recent works such as DAPO and Dr.GRPO highlighted limitations of this design and accordingly advocated *token aggregation*, which directly averages the clipped objective over all tokens in the sampled group, as a better alternative (Yu et al., 2025; Liu et al., 2025b). In this paper, we show that these two rules induce systematically different optimization biases and can lead to substantially different training dynamics and final performance.

We show that token aggregation introduces a *sign-length coupling bias*: the relative contribution of positive and negative samples on the policy gradient depends not only on their normalized advantages, but also on their average response lengths. Therefore, when positive and negative responses have different length distributions, token aggregation can systematically amplify one side of the update.

Sequence aggregation removes this positive-negative length coupling by assigning equal weight to each response. However, this introduces a

different bias: longer responses are implicitly downweighted because each sequence contributes equally regardless of how many tokens it contains (Yu et al., 2025; Liu et al., 2025b).

These two biases matter in practice. We find that token aggregation can be favorable when response length variance is large, since it avoids overly suppressing long responses. However, it is also more sensitive to positive-negative length imbalance and often leads to less stable optimization. This tension suggests that a better aggregation rule should preserve the sign-balance property of sequence aggregation without inheriting its strong sequence-level equal-weighting effect.

To this end, we propose **Balanced Aggregation (BA)**. The key idea is simple: we first split responses within each group into positive and negative subsets according to the sign of their normalized advantages, compute token-level means separately within each subset, and then combine the two subset losses using weights proportional to the number of sequences in each subset. This construction removes the positive-negative length coupling induced by token aggregation, while retaining token-level averaging within each sign group. As a result, BA preserves the same inter-sign balancing principle as sequence aggregation, but does not force every response to have equal weight within a sign group.

We evaluate BA on GRPO-style RLVR training using Qwen2.5-Math-7B and Qwen3-1.7B across DAPO and Polaris training sets, and report results on six evaluation benchmarks including Math-500, AIME 2024, AIME 2025, OlympicBench, Minerva-MATH, and LiveCodeBench. Across both weak and strong model regimes, BA consistently delivers stronger final performance and better training stability than standard token and sequence aggregation. Our analysis further shows that the relative effectiveness of token and sequence aggregation can be largely explained by two factors: response length variance and the response-length gap between positive and negative samples.

Our contributions are as follows:

- We show that loss aggregation in GRPO-style RLVR is not a benign implementation detail, and provide a unified analysis of the sign-length coupling bias in token aggregation and the sequence equal-weighting bias in sequence aggregation.
- We propose Balanced Aggregation (BA), a simple drop-in replacement that performs token-level averaging separately within the positive and negative subsets before combining them, thereby avoiding the main bias of token aggregation without imposing the strong equal-weighting effect of sequence aggregation.
- We provide extensive empirical evidence that BA improves robustness and final performance across models, datasets, and evaluation benchmarks, and clarify when token aggregation or sequence aggregation is preferable.

2 Related Work

RLVR and GRPO-Style Post-Training Recent progress in reasoning-oriented LLM post-training has highlighted the importance of reinforcement learning, as reflected by the success of systems such as OpenAI's o1, DeepSeek-R1 (OpenAI et al., 2024; Guo et al., 2025; Zeng et al., 2024). In tasks with programmatically verifiable outcomes, reinforcement learning with verifiable rewards (RLVR) has emerged as a particularly attractive paradigm because it avoids learned reward modeling and provides a scalable training signal for reasoning and code generation. On the optimization side, PPO (Schulman et al., 2017) has long served as the standard policy optimization backbone, while GRPO, introduced in DeepSeekMath (Shao et al., 2024), further reduces training cost by replacing the critic with group-relative reward normalization. This critic-free formulation has made GRPO-style training a practical foundation for large-scale RLVR.

RLVR Training Tricks A growing line of work has improved RLVR training from multiple angles. Some methods focus on reducing train-infer mismatch and improving training stability (Yao et al., 2025; Liu et al., 2025a; Ma et al., 2025; DeepSeek-AI et al., 2025; Zhiyuan et al., 2025). Another line studies clipping and trust-region design, including asymmetric clipping (Yu et al., 2025) and soft clipping (MiniMax et al., 2025). A related direction directly improves importance sampling, with methods such as GSPO and ASPO (Zheng et al., 2025; Wang et al., 2025). Some works also examine the role of advantage estimation and normalization, showing that the standard-deviation normalization can introduce nontrivial optimization bias (Liu et al., 2025b; DeepSeek-AI et al., 2025). Recent empirical studies further revisit a wide range

of RLVR training heuristics and scaling choices, highlighting that many commonly used tricks can have subtle effects (Liu et al., 2025c; Khatri et al., 2025). Together, these studies show that RLVR performance depends heavily on a collection of low-level optimization choices rather than on the policy objective alone.

Aggregation in GRPO-Style RL Among these design choices, how token-level policy gradient terms are aggregated has received less systematic attention. Standard GRPO uses sequence aggregation, which first averages token-level contributions within each response and then averages across responses. Recent works such as DAPO and Dr.GRPO identified limitations of this design in long-form reasoning and accordingly advocated token-level alternatives (Yu et al., 2025; Liu et al., 2025b). GMPO improves optimization stability in a different way, by replacing the arithmetic mean with a geometric mean (Zhao et al., 2025). By contrast, our focus is the bias induced by the aggregation rule itself, so we center the analysis and experiments on sequence aggregation and token aggregation, and position Balanced Aggregation as a simple alternative that directly addresses their respective biases.

3 Method

3.1 GRPO-style RLVR

Given a prompt x, GRPO-style RLVR samples a group of G responses from the current policy,

$$o_1, o_2, \ldots, o_G \sim \pi_\theta(\cdot \mid x). \tag{1}$$

Each response receives a verifiable scalar reward r_i, and rewards are normalized within the group to form sequence-level advantages:

$$\mu = \frac{1}{G}\sum_{i=1}^{G} r_i, \qquad \sigma = \sqrt{\frac{1}{G}\sum_{i=1}^{G}(r_i-\mu)^2 + \epsilon}, \tag{2}$$

$$\hat{A}_i = \frac{r_i - \mu}{\sigma}. \tag{3}$$

The key point for our analysis is that $\hat{A}_i$ is shared by all tokens in response o_i.

3.2 Token-Level PPO Objective

Let T_i denote the length of response o_i. For token t, define

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid x, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid x, o_{i,<t})}. \tag{4}$$

The clipped token-level PPO contribution is

$$\phi_{i,t}(\theta) = \min\Big(\rho_{i,t}(\theta)\hat{A}_i, \; \text{clip}(\rho_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_i\Big). \tag{5}$$

The aggregation rule specifies how the token-level terms $\phi_{i,t}$ are reduced to a group-level objective.

3.3 Aggregation rules

Two common reductions are token aggregation and sequence aggregation. Token aggregation averages over all generated tokens:

$$\mathcal{J}_{\text{token}}(\theta) = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{G}\sum_{t=1}^{T_i}\phi_{i,t}(\theta)\right], \qquad N = \sum_{i=1}^{G} T_i. \tag{6}$$

Sequence aggregation first averages within each response and then averages across responses:

$$\mathcal{J}_{\text{seq}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i}\phi_{i,t}(\theta)\right]. \tag{7}$$

Both use the same token-level objective, but they induce different length-dependent weights.

3.4 Motivation: Aggregation Bias in GRPO

To expose the bias, partition responses by the sign of their advantage:

$$S_+ = \{i \mid \hat{A}_i > 0\}, \qquad S_- = \{i \mid \hat{A}_i < 0\}. \tag{8}$$

Let $k = |S_+|$, so $|S_-| = G - k$. For analysis, we factor the sequence-level advantage out of the token-level PPO contribution and define $\delta_{i,t}(\theta) := \phi_{i,t}(\theta)/\hat{A}_i$ for nonzero-advantage responses:

$$\phi_{i,t}(\theta) = \hat{A}_i\, \delta_{i,t}(\theta). \tag{9}$$

Under binary rewards, group normalization gives

$$\hat{A}_i = \begin{cases} \sqrt{\frac{G-k}{k}}, & i \in S_+, \\ -\sqrt{\frac{k}{G-k}}, & i \in S_-. \end{cases} \tag{10}$$

Substituting this into token aggregation yields

$$\mathcal{J}_{\text{token}} \propto \frac{\sqrt{k(G-k)}}{N}\left(\bar{T}_+\,\bar{\delta}_+^{\text{tok}} - \bar{T}_-\,\bar{\delta}_-^{\text{tok}}\right), \tag{11}$$

where

$$\bar{T}_+ = \frac{1}{k}\sum_{i\in S_+} T_i, \qquad \bar{T}_- = \frac{1}{G-k}\sum_{i\in S_-} T_i, \tag{12}$$

and $\bar{\delta}^{\text{tok}}_{\pm}$ denote token-level averages within each sign group. More explicitly, with $N_+ = \sum_{i\in S_+} T_i$ and $N_- = \sum_{i\in S_-} T_i$,

$$\bar{\delta}^{\text{tok}}_{\pm} = \frac{1}{N_\pm}\sum_{i\in S_\pm}\sum_{t=1}^{T_i}\delta_{i,t}. \tag{13}$$

Thus token aggregation creates a **sign-length coupling bias**: when $\bar{T}_+ \neq \bar{T}_-$, the positive and negative gradient contributions are reweighted by their average lengths. This is not merely a normalization detail; it changes the effective balance between reward-increasing and reward-decreasing updates whenever response length correlates with reward sign.

By contrast, sequence aggregation removes this cross-sign length coupling:

$$\mathcal{J}_{\text{seq}} \propto \frac{\sqrt{k(G-k)}}{G}\left(\bar{\delta}^{\text{seq}}_+ - \bar{\delta}^{\text{seq}}_-\right), \tag{14}$$

where

$$\begin{aligned}\bar{\delta}^{\text{seq}}_+ &= \frac{1}{k}\sum_{i\in S_+}\left(\frac{1}{T_i}\sum_{t=1}^{T_i}\delta_{i,t}\right),\\ \bar{\delta}^{\text{seq}}_- &= \frac{1}{G-k}\sum_{i\in S_-}\left(\frac{1}{T_i}\sum_{t=1}^{T_i}\delta_{i,t}\right).\end{aligned} \tag{15}$$

This equalizes the positive and negative signs at the group level, but introduces a **sequence equal-weighting bias**: each response has equal weight regardless of its token count. This can be desirable when the dominant issue is sign-length coupling, but it may discard useful token-level averaging when response lengths vary substantially for reasons unrelated to correctness. The goal of BA is to preserve the sign balance of sequence aggregation while retaining token-level averaging within each sign group.

3.5 Balanced Aggregation

We propose **Balanced Aggregation (BA)**, a simple aggregation rule that separates positive and negative samples before averaging.

We first compute token-level mean losses within the positive and negative subsets:

$$\begin{aligned}\mathcal{L}_+ &= \frac{1}{N_+}\sum_{i\in S_+}\sum_{t=1}^{T_i}\phi_{i,t}(\theta),\\ \mathcal{L}_- &= \frac{1}{N_-}\sum_{i\in S_-}\sum_{t=1}^{T_i}\phi_{i,t}(\theta).\end{aligned} \tag{16}$$

We then combine them using sequence-count-based weights:

$$\mathcal{J}_{\text{BA}}(\theta) = \mathbb{E}\left[\frac{k}{G}\mathcal{L}_+ + \frac{G-k}{G}\mathcal{L}_-\right]. \tag{17}$$

Equivalently,

$$\begin{aligned}\mathcal{J}_{\text{BA}}(\theta) =& \mathbb{E}\Bigg[\frac{k}{G}\cdot\frac{1}{N_+}\sum_{i\in S_+}\sum_{t=1}^{T_i}\phi_{i,t}(\theta)\\ &+ \frac{G-k}{G}\cdot\frac{1}{N_-}\sum_{i\in S_-}\sum_{t=1}^{T_i}\phi_{i,t}(\theta)\Bigg].\end{aligned} \tag{18}$$

The intuition is straightforward. Within each sign group, BA retains token-level averaging, so it does not force every response to have equal weight. Across sign groups, BA uses sequence-count-based reweighting, which restores the same positive-negative balancing principle as sequence aggregation. In particular, the weights k/G and $(G-k)/G$ are chosen so that, under the binary-reward GRPO setting, BA induces the same inter-sign prefactor as sequence aggregation; a short derivation is provided in Appendix A.

3.6 Connection to Sequence Aggregation

BA is closely related to sequence aggregation, but the two are not equivalent.

Under the same binary-reward GRPO setting, substituting the normalized advantages into BA yields

$$\mathcal{J}_{\text{BA}} \propto \frac{\sqrt{k(G-k)}}{G}\left(\bar{\delta}^{\text{BA}}_+ - \bar{\delta}^{\text{BA}}_-\right), \tag{19}$$

where

$$\begin{aligned}\bar{\delta}^{\text{BA}}_+ &= \frac{1}{N_+}\sum_{i\in S_+}\sum_{t=1}^{T_i}\delta_{i,t},\\ \bar{\delta}^{\text{BA}}_- &= \frac{1}{N_-}\sum_{i\in S_-}\sum_{t=1}^{T_i}\delta_{i,t}.\end{aligned} \tag{20}$$

By contrast, sequence aggregation has exactly the same inter-sign form as in Eq. (14), with within-sign averages defined in Eq. (15).

Therefore, BA and sequence aggregation share the same *inter-sign balancing* structure: both remove the sign-length coupling of token aggregation and induce the same positive-negative prefactor $\sqrt{k(G-k)}/G$. However, they differ in their *within-sign averaging* rule. Sequence aggregation gives equal weight to each response within a sign group, whereas BA averages over all tokens within that sign group.

In general,

$$\bar{\delta}_{\pm}^{\text{seq}} \neq \bar{\delta}_{\pm}^{\text{BA}}, \tag{21}$$

unless all responses within a sign group have the same length. Thus, BA should be understood as preserving the sign-balance property of sequence aggregation without inheriting its strong per-sequence equal-weighting effect.

BA is a simple drop-in replacement for the aggregation step in GRPO-style RLVR. It removes the sign-length coupling bias of token aggregation while avoiding the strong sequence equal-weighting bias of sequence aggregation. Although the current formulation of BA is derived under the binary-reward setting, BA can naturally extend to non-binary rewards, which is shown in Appendix B.

4 Experiments

4.1 Experimental Settings

Training Data We conduct RLVR training on two datasets: **DAPO-17k** (approximately 17,000 mathematical reasoning problems) and **Polaris** (approximately 53,000 mathematical problems) (Yu et al., 2025; Group et al., 2025). Both consist of problem-answer pairs, where answers are used to compute verifiable rewards for generated responses.

Evaluation Benchmarks We evaluate on six benchmarks covering both difficult reasoning and coding tasks: Math-500, AIME-2024, AIME-2025, OlympicBench, Minerva-MATH, and LivecodeBench (Lightman et al., 2023; Lewkowycz et al., 2022; He et al., 2024; Jain et al., 2024).

Compared Methods We compare three aggregation rules applied within the DAPO algorithm:

- **token-agg**: Token-level averaging, where the clipped PPO objective is averaged over all tokens in the sampled group. This is used in DAPO and Dr.GRPO (Yu et al., 2025; Liu et al., 2025b).
- **seq-agg**: Sequence-level averaging, where token-level contributions are first averaged within each response and then averaged across responses. This is the default in GRPO (Shao et al., 2024).
- **balanced-agg**: Our proposed balanced aggregation, which splits responses by advantage sign, computes token-level means separately within positive and negative subsets, and combines them with sequence-count-based weights.

All other components (advantage normalization, PPO clipping, sampling) are kept identical across methods.

Training Details We train **Qwen2.5-Math-7B** and **Qwen3-1.7B** (Yang et al., 2024, 2025) with maximum response lengths of 2,048 and 8,192 tokens. Training is implemented in the verl framework, using group size $G = 16$, learning rate 10^{-6}, and 500 total steps. We apply PPO clipping bounds of 0.2 and 0.28 (the standard DAPO setting). The global batch size is 128 prompts, each generating 16 responses via vLLM (temperature 1.0). Other hyper-parameters follow the standard DAPO configuration (Yu et al., 2025).

Evaluation Protocol We sample 8 responses per prompt (temperature 1.0). For math benchmarks, correctness is determined using OpenCompass's rule-based verifier (Team, 2024); for LivecodeBench, we execute the generated code against unit tests (Jain et al., 2024). We report three metrics: **peak accuracy** (highest accuracy observed during training), **peak best@8 accuracy**, and **last-step accuracy** (accuracy at the final training step).

4.2 Main Results

To evaluate **Balanced Aggregation** (denoted as **balanced-agg**), we compare it against **token-agg** and **seq-agg** baselines on two training datasets: DAPO-17k and Polaris. We benchmark the aggregation methods on two base models: Qwen2.5-Math-7B and Qwen3-1.7B.

Table 1: Main experimental results on DAPO-17k and Polaris datasets using Qwen2.5-Math-7B and Qwen3-1.7B models. We report the average peak and last-step accuracy across six evaluation benchmarks.

			Peak Performance		Last-Step	
Dataset	**Model**	**Aggregation**	**Acc@8**	**Best@8**	**Acc@8**	**Best@8**
DAPO-17k	*Qwen2.5-Math-7B*	seq-agg	0.3578	0.4821	**0.3446**	0.4598
		token-agg	0.3595	0.4869	0.3364	0.4434
		balanced-agg	**0.3634**	**0.4896**	0.3424	**0.4627**
	Qwen3-1.7B	seq-agg	**0.4947**	0.5990	0.4481	0.5522
		token-agg	0.4876	0.6058	0.4360	0.5608
		balanced-agg	0.4928	**0.6086**	**0.4695**	**0.5861**
Polaris	*Qwen2.5-Math-7B*	seq-agg	0.3405	0.4697	0.3172	0.4360
		token-agg	**0.3539**	**0.4768**	0.3292	**0.4527**
		balanced-agg	0.3423	0.4750	**0.3319**	0.4443
	Qwen3-1.7B	seq-agg	0.4812	0.5950	0.4614	**0.5730**
		token-agg	0.4842	0.6053	0.4349	0.5557
		balanced-agg	**0.4879**	**0.6094**	**0.4640**	0.5550

Table 1 presents the average scores across six evaluation benchmarks. Since a full breakdown would necessitate an overly large table, detailed per-benchmark results are shown in Figure 1. Furthermore, because RLVR training dynamics can be highly volatile in later stages, the highest peak performance does not guarantee the best last-step performance. We therefore explicitly report both **peak** and **last-step** metrics to comprehensively evaluate each method's training stability.

Overall results on DAPO-17k For Qwen2.5-Math-7B, **token-agg** yields better peak performance than **seq-agg**, but **balanced-agg** surpasses both to establish the highest peak metrics. For Qwen3-1.7B, the relationship flips: **seq-agg** becomes superior to **token-agg**. This is highly relevant since **token-agg** is the default in frameworks like *verl*, yet it is clearly not universally better than seq-agg. More crucially, while **balanced-agg** achieves peak metrics comparable to **seq-agg**, it successfully prevents the severe degradation often observed in later stages of RLVR, maintaining much higher last-step accuracies.

Overall results on Polaris Similar performance dynamics are observed on Polaris. For Qwen2.5-Math-7B, **token-agg** exhibits the highest peak metrics but suffers noticeable degradation toward the end of training. In contrast, **balanced-agg** achieves the most robust last-step accuracy and strictly outperforms **seq-agg** across all evaluated metrics. For Qwen3-1.7B, **balanced-agg** achieves the highest peak metrics compared to both baselines. In the final training stages, while **token-agg** suffers a severe collapse, both **seq-agg** and **balanced-agg** maintain much more robust last-step accuracies.

Cross-setting summary Across both datasets, a consistent dynamic emerges: **token-agg** performs better on Qwen2.5-Math-7B, whereas **seq-agg** is more stable and accurate on Qwen3-1.7B. This suggests neither standard aggregation provides a consistently reliable optimization signal across different base models. We delve into the reasons behind this performance flip in Section 4.3. **Balanced-agg** successfully bridges this gap, consistently ranking as the best or highly competitive method across our evaluated models and datasets. Its ability to simultaneously preserve within-sign token-level averaging while removing the positive-negative length coupling substantially improves training stability.

4.3 Analysis

The main results in Table 1 show that aggregation rules interact strongly with the base model and training corpus, and that peak accuracy alone can be misleading when training is volatile. In this subsection, we unpack these findings along three complementary axes. First, we compare peak versus last-step accuracy at the per-benchmark level to make training stability visually explicit. Second, we connect the observed optimization behavior to our theoretical account by examining policy-gradient loss trajectories during training. Finally, we connect the theory in Section 3.4 to the model-dependent flip in Section 4.2 and to length statistics over training (Figure 3).

4.3.1 Peak vs. Last-Step Performance

Figure 1 compares peak and last-step accuracy on each benchmark, where each bar is averaged

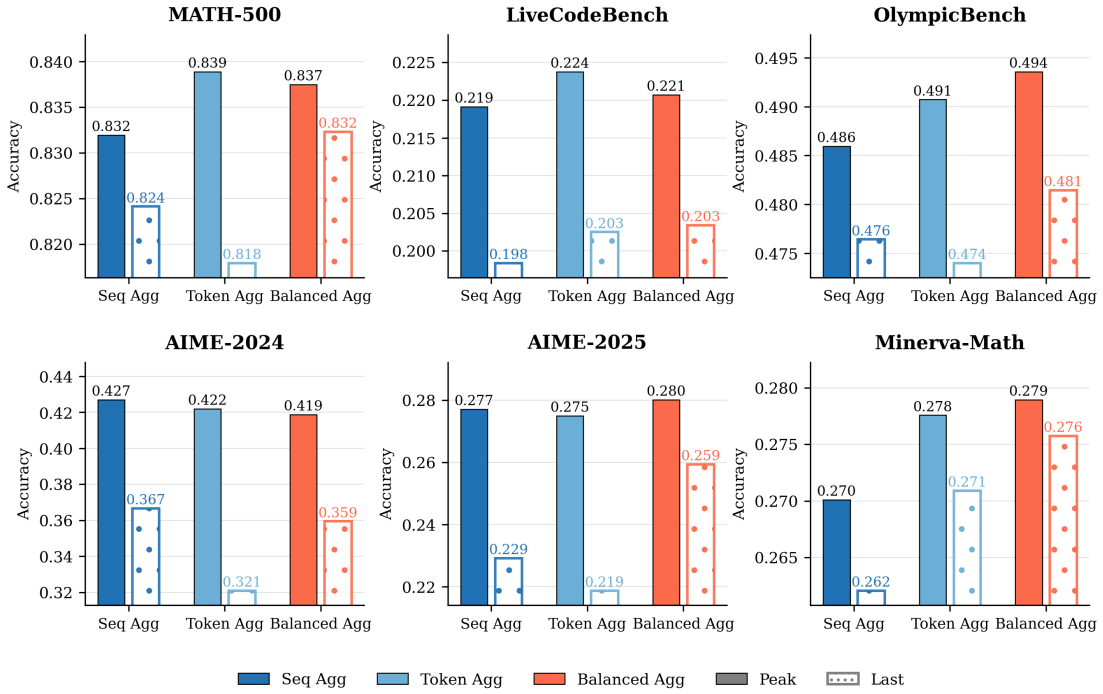


Figure 1: Comparison of peak and last-step accuracy across evaluation benchmarks. For each benchmark, values are averaged over four training settings: Qwen2.5-Math-7B and Qwen3-1.7B, each trained on DAPO-17k and Polaris. Solid bars indicate peak performance, while dashed bars indicate last-step performance.

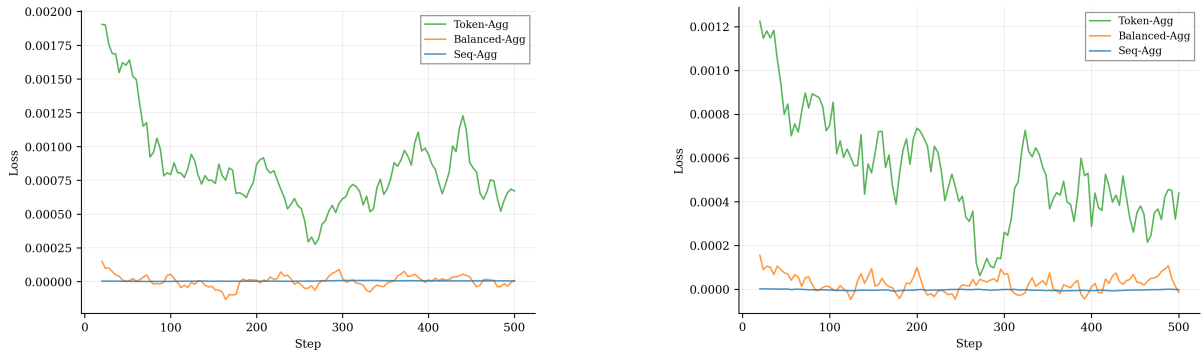
over four training settings: Qwen2.5-Math-7B and Qwen3-1.7B, each trained on DAPO-17k and Polaris. At peak performance, **token-agg** and **balanced-agg** are very close, and on most benchmarks both outperform **seq-agg**. The difference emerges at the final checkpoint: **token-agg** exhibits the largest peak-to-last drop on nearly all benchmarks, whereas **balanced-agg** preserves its gains much better and achieves the best or tied-best last-step result on five of the six benchmarks. The largest peak-to-last gaps appear on AIME-2024 and AIME-2025, likely due in part to the higher variance of their small evaluation sets. Overall, these results further indicate that **balanced-agg** delivers substantially stronger training stability than standard token aggregation while remaining highly competitive at peak performance.

4.3.2 Policy-gradient loss dynamics and aggregation bias

To empirically validate our theoretical observations regarding aggregation bias, we analyze the training dynamics of the policy-gradient loss. Figure 2 plots the evolution of this loss over the course of training. We observe a stark contrast between standard token-level aggregation and the sequence-balanced configurations: the loss for **token-agg** experiences massive drifts and oscillates consistently in a region far above zero. In contrast, both **seq-agg** and **balanced-agg** remain highly stable, with their loss trajectories oscillating tightly around zero. These empirical dynamics directly corroborate our earlier analysis of the positive-negative sign-length coupling bias. This behavior directly matches our earlier theory: when the positive-negative response-length gap is large, **token-agg** suffers from sign-length coupling, which skews the effective gradient balance and produces a persistent drift away from zero. By contrast, **seq-agg** and **balanced-agg** remove this coupling, so their loss trajectories remain much more stable and close to zero.

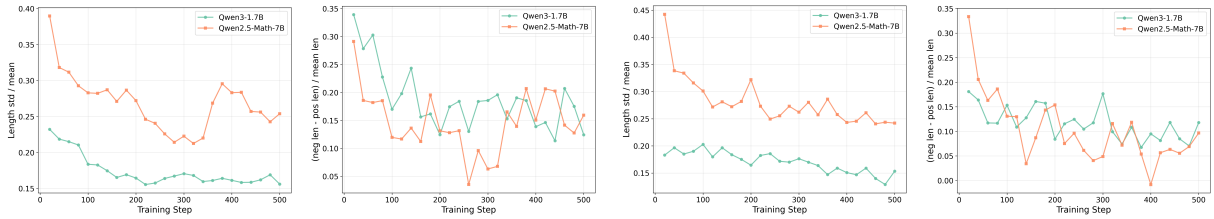
4.3.3 Understanding the Model-Dependent Flip

Our theory in Sections 3.3 and 3.4 suggests a simple criterion for when each aggregation rule should be preferred. **Token-agg** should be more favorable when response-length variation is large but the positive–negative length gap is relatively mild, because in that regime the weakness of **seq-agg** is more pronounced than the weakness of **token-agg**. Conversely, **seq-agg** should be more favorable when response-length variation is small but the positive–negative length gap is large, because in



(a) Qwen2.5 on DAPO (b) Qwen2.5 on Polaris

Figure 2: Training dynamics of the policy-gradient loss for Qwen2.5-Math-7B on DAPO-17k and Polaris. Triggered by the phenomenon that negative responses are systematically longer than positive responses, **token-agg** places disproportionate weight on negative samples, causing its loss to severely deviate from zero. In contrast, **seq-agg** and **balanced-agg** actively decouple this sign-length bias and remain much more stable.



(a) DAPO: length CV (b) DAPO: pos-neg gap (c) Polaris: length CV (d) Polaris: pos-neg gap

Figure 3: Length-distribution statistics related to the relative behavior of token and sequence aggregation. We report response-length CV, where CV denotes the coefficient of variation, and the pos-neg gap, defined as the average length of positive responses minus that of negative responses, over training on DAPO-17k and Polaris.

that regime the sign-length coupling bias in **token-agg** becomes the dominant issue. In Section 4.2, we observe exactly such a model-dependent flip: **token-agg** performs better on Qwen2.5-Math-7B, whereas **seq-agg** performs better on Qwen3-1.7B. Figure 3 tests this explanation by tracking over training the two quantities highlighted by the theory on DAPO-17k and Polaris.

When is Token-Agg Better than Seq-Agg? Qwen2.5-Math-7B falls into the regime favorable to **token-agg**. As shown in Figure 3, it exhibits substantially larger response-length variation than Qwen3-1.7B, while its positive–negative length gap is generally milder. This means the sequence-level equal-weighting effect in **seq-agg** is more harmful than the sign-length coupling bias in **token-agg**, which explains why **token-agg** can outperform **seq-agg** on Qwen2.5-Math-7B in Table 1.

When is Seq-Agg Better than Token-Agg? Qwen3-1.7B shows the opposite pattern. Its response-length variation is much smaller, so the weakness of **seq-agg** is reduced. Meanwhile, its positive–negative length gap is consistently larger, making the sign-length coupling bias in **token-agg** more severe. This is why **seq-agg** outperforms **token-agg** on Qwen3-1.7B in Table 1. The same comparison also helps explain why **balanced-agg** is robust across both regimes: it mitigates the positive–negative coupling that hurts **token-agg** without paying the full cost of sequence-level equal weighting.

5 Conclusion

We studied how aggregation in GRPO-style RLVR shapes optimization. Token aggregation suffers from sign-length coupling, while sequence aggregation removes this coupling but imposes strong per-sequence equal weighting. Balanced Aggregation (BA) preserves sign balance while retaining token-level averaging within each sign group. Across models, datasets, and benchmarks, BA improves training stability and final performance, showing that aggregation is a first-class design choice.

Limitations

Our study focuses on RLVR settings with verifiable rewards, primarily mathematical reasoning and code generation. These settings are a natural testbed for aggregation bias because rewards are discrete and sequence-level, but further work is needed to examine whether the same trade-offs appear in more open-ended preference-based or multi-objective training.

Our experiments cover two base models and two training corpora, with evaluation across several reasoning and coding benchmarks. While this is sufficient to expose the model-dependent flip between token and sequence aggregation, broader experiments with larger model families, additional RLVR datasets, and different rollout budgets would further test the generality of the observed regimes.

Finally, our analysis isolates length-related aggregation effects. Other factors in RLVR training, such as reward noise, sampling temperature, KL control, and optimization hyperparameters, may interact with aggregation choices. We leave a more exhaustive study of these interactions to future work.

References

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 245 others. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *Preprint*, arXiv:2512.02556.

HKU NLP Group, ByteDance Seed, and Fudan University. 2025. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Annual Meeting of the Association for Computational Linguistics*.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida I. Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S. Dhillon, David Brandfonbrener, and Rishabh Agarwal. 2025. The art of scaling reinforcement learning compute for llms. *Preprint*, arXiv:2510.13786.

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

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *Preprint*, arXiv:2305.20050.

Jiacai Liu, Yingru Li, Yuqian Fu, Jiawei Wang, Qian Liu, and Yu Shen. 2025a. When speed kills stability: Demystifying RL collapse from the training-inference mismatch.

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

Zihe Liu, Jiashun Liu, Yancheng He, Weixun Wang, Jiaheng Liu, Ling Pan, Xinyu Hu, Shaopan Xiong, Ju Huang, Jian Hu, Shengyi Huang, Johan Obando-Ceron, Siran Yang, Jiamang Wang, Wenbo Su, and Bo Zheng. 2025c. Part i: Tricks or traps? a deep dive into rl for llm reasoning. *Preprint*, arXiv:2508.08221.

Wenhan Ma, Hailin Zhang, Liang Zhao, Yifan Song, Yudong Wang, Zhifang Sui, and Fuli Luo. 2025. Stabilizing moe reinforcement learning by aligning training and inference routers. *Preprint*, arXiv:2510.11370.

MiniMax, :, Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, Chengjun Xiao, Chengyu Du, Chi Zhang, Chu Qiao, Chunhao Zhang, Chunhui Du, Congchao Guo, and 109 others. 2025. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *Preprint*, arXiv:2506.13585.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, and 244 others. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

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

Yiping Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, Daya Guo, Dejian Yang, Dejian Yang, and Ruoyu Zhang. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

OpenCompass Team. 2024. Opencompass: A universal evaluation platform for foundation models. *Preprint*, arXiv:2410.16256.

Jiakang Wang, Runze Liu, Lei Lin, Wenping Hu, Xiu Li, Fuzheng Zhang, Guorui Zhou, and Kun Gai. 2025. Aspo: Asymmetric importance sampling policy optimization. *Preprint*, arXiv:2510.06062.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 42 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. 2025. Your efficient rl framework secretly brings you off-policy rl training.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Bo Wang, Shimin Li, Yunhua Zhou, Qipeng Guo, Xuanjing Huang, and Xipeng Qiu. 2024. Scaling of search and learning: A roadmap to reproduce o1 from reinforcement learning perspective. *Preprint*, arXiv:2412.14135.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, Fang Wan, and Furu Wei. 2025. Geometric-mean policy optimization. *Preprint*, arXiv:2507.20673.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2025. Group sequence policy optimization. *Preprint*, arXiv:2507.18071.

Zeng Zhiyuan, Jiashuo Liu, Zhangyue Yin, Ge Zhang, Wenhao Huang, and Xipeng Qiu. 2025. Rloop: An self-improving framework for reinforcement learning with iterative policy initialization. *Preprint*, arXiv:2511.04285.

Appendix A Why Use Sequence-Count Weights in BA?

Here we briefly justify the choice of weights k/G and $(G-k)/G$ in BA. Recall that under the binary-reward GRPO setting, the normalized advantages are

$$\hat{A}_i = \begin{cases} \sqrt{\frac{G-k}{k}}, & i \in S_+, \\ -\sqrt{\frac{k}{G-k}}, & i \in S_-. \end{cases} \tag{22}$$

Substituting these values into the BA objective gives

$$\begin{aligned} \mathcal{J}_{\mathrm{BA}} =& \frac{k}{G}\left(\sqrt{\frac{G-k}{k}}\,\bar{\delta}_+^{\mathrm{BA}}\right) \\ &+ \frac{G-k}{G}\left(-\sqrt{\frac{k}{G-k}}\,\bar{\delta}_-^{\mathrm{BA}}\right). \end{aligned} \tag{23}$$

Rearranging yields

$$\mathcal{J}_{\mathrm{BA}} \propto \frac{\sqrt{k(G-k)}}{G}\left(\bar{\delta}_+^{\mathrm{BA}} - \bar{\delta}_-^{\mathrm{BA}}\right), \tag{24}$$

which is exactly the same inter-sign prefactor as sequence aggregation in Eq. (14). Therefore, the sequence-count weights k/G and $(G-k)/G$ are not arbitrary: they are precisely what ensures that BA preserves the same positive-negative balancing principle as sequence aggregation, while still using token-level averaging within each sign group.

Appendix B Extension to Non-Binary Rewards

Our main presentation of Balanced Aggregation (BA) focuses on the standard binary-reward GRPO setting, where the normalized advantages are constant within each sign subset. In that case, the positive and negative groups can be characterized solely by their sequence counts, which leads to the simple sequence-count weights k/G and $(G-k)/G$.

For general real-valued rewards, however, the normalized advantages $\hat{A}_i$ are no longer constant even within the same sign subset. As a result, sequence counts alone are no longer sufficient to characterize the relative contribution of the positive and negative subsets. In particular, if we write the token-level PPO contribution as

$$\phi_{i,t}(\theta) = \hat{A}_i\,\delta_{i,t}(\theta),$$

then the total contribution of each sign subset depends not only on the number of responses and their lengths, but also on the magnitudes of their advantages.

To generalize BA to this setting, we first define the positive and negative subsets

$$S_+ = \{i \mid \hat{A}_i > 0\}, \qquad S_- = \{i \mid \hat{A}_i < 0\}.$$

We then define the corresponding sign-wise advantage masses

$$M_+ = \sum_{i\in S_+} \hat{A}_i, \qquad M_- = \sum_{i\in S_-} (-\hat{A}_i),$$

and the sign-wise advantage-weighted token masses

$$Z_+ = \sum_{i\in S_+} \hat{A}_i T_i, \qquad Z_- = \sum_{i\in S_-} (-\hat{A}_i) T_i.$$

Using these quantities, we define the generalized BA objective as

$$\begin{aligned} \mathcal{J}_{\text{BA-gen}}(\theta) =& \mathbb{E}\Bigg[\frac{M_+}{G}\cdot\frac{1}{Z_+}\sum_{i\in S_+}\sum_{t=1}^{T_i}\phi_{i,t}(\theta) \\ &+ \frac{M_-}{G}\cdot\frac{1}{Z_-}\sum_{i\in S_-}\sum_{t=1}^{T_i}\phi_{i,t}(\theta)\Bigg]. \end{aligned}$$

Since $\phi_{i,t}(\theta)$ is positive-advantage-weighted on S_+ and negative-advantage-weighted on S_-, this construction preserves the original policy-gradient weighting induced by $\hat{A}_i$, while normalizing the positive and negative subsets separately. Equivalently, substituting $\phi_{i,t}(\theta) = \hat{A}_i\delta_{i,t}(\theta)$ gives

$$\mathcal{J}_{\text{BA-gen}}(\theta) = \mathbb{E}\left[\frac{M_+}{G}\,\bar{\delta}_+^{\mathrm{gen}} - \frac{M_-}{G}\,\bar{\delta}_-^{\mathrm{gen}}\right],$$

where

$$\begin{aligned} \bar{\delta}_+^{\mathrm{gen}} &= \frac{1}{Z_+}\sum_{i\in S_+}\hat{A}_i\sum_{t=1}^{T_i}\delta_{i,t}(\theta), \\ \bar{\delta}_-^{\mathrm{gen}} &= \frac{1}{Z_-}\sum_{i\in S_-}(-\hat{A}_i)\sum_{t=1}^{T_i}\delta_{i,t}(\theta). \end{aligned}$$

Therefore, the generalized BA retains the same core principle as the binary version: positive and negative samples are first normalized within their own sign subsets and are then recombined in a sign-balanced manner. The key difference is that, in the non-binary case, sign balance is determined by advantage mass rather than sequence count.

Moreover, under the usual group-normalization condition

$$\sum_{i=1}^{G} \hat{A}_i = 0,$$

we have

$$M_+ = M_- = \frac{1}{2} \sum_{i=1}^{G} |\hat{A}_i|.$$

Hence, the positive and negative subsets remain symmetric at the inter-sign level, so the generalized BA still removes the cross-sign sign-length coupling induced by standard token aggregation.

Finally, under binary rewards, $\hat{A}_i$ is constant within each sign subset. In that case,

$$\begin{aligned} M_+ &= k a_+, \qquad M_- = (G-k) a_-, \\ Z_+ &= a_+ N_+, \quad Z_- = a_- N_-, \end{aligned}$$

for some constants $a_+ > 0$ and $a_- > 0$. Substituting these expressions into $\mathcal{J}_{\text{BA-gen}}$ recovers exactly the original BA objective defined in the main text. Therefore, the generalized formulation is a strict extension of BA rather than a different objective.