AdaDPO: Self-Adaptive Direct Preference Optimization with Balanced Gradient Updates

Shaolong Chen
Incept Labs
Houston, TX

Madalina Ciobanu
Incept Labs
Houston, TX

Qingqing Mao*
Incept Labs, Houston, TX
Titan Holdings, San Francisco, CA
qmao@inceptlabs.ai

Ritankar Das
Incept Labs, Houston, TX
Titan Holdings, San Francisco, CA

Abstract

Direct Preference Optimization (DPO) is an *offline* decision-making algorithm: a policy is trained entirely from a fixed preference dataset, with no online interaction. Recent theoretical analysis uncovers an asymmetric gradient pathology in DPO that misallocates this offline learning signal: the loss suppresses dispreferred responses substantially faster than it promotes preferred ones, so the model predominantly learns to avoid bad answers rather than to generate good ones. We propose **AdaDPO**, a Self-**Ada**ptive variant of the **D**irect **P**reference **O**ptimization algorithm that introduces per-preference-pair, stop-gradient-based coefficients derived directly from the policy model's generation probabilities. AdaDPO is constructed to balance gradient magnitudes between preferred and dispreferred probabilities; the practical implementation balances per-token gradients and applies a numerical clipping bound for stability, while retaining DPO's original hyperparameter structure. In preliminary experiments on Llama-3-8B-Instruct trained on UltraFeedback, AdaDPO consistently outperforms DPO on AlpacaEval 2: it achieves higher length-controlled win rates (LC) in 81% of hyperparameter combinations and enlarges the LC-over-WR margin in 88% of combinations, indicating effective mitigation of length bias. Because it operates purely at the loss level, AdaDPO is a drop-in correction for preference-based alignment pipelines.

1 Introduction

Direct Preference Optimization (DPO) [1] aligns LLMs with human preferences by training a policy directly from a fixed preference dataset, replacing the reward model and online RL loop of RLHF [2–4]. Thus, DPO is an *offline* decision-making algorithm: a policy is trained entirely from logged data with no online interaction. However, DPO suffers from an asymmetric gradient pathology [5]: the loss applies a substantially larger gradient to dispreferred vs. preferred responses; DPO-trained models learn to suppress bad answers rather than promote good ones. This is structural (the same coefficient β multiplies both log-ratios), and existing variants (SimPO [6], R-DPO [7], IPO [8], CPO [9], ORPO [10]) do not address it. In an offline setting, this imbalance directly misallocates the limited learning signal available.

We propose AdaDPO, a self-adaptive variant of DPO that replaces the fixed β with per-preference-pair coefficients β_w and β_l. Setting $\beta_l = \beta$ and computing β_w via a stop-gradient-based ratio of generation probabilities yields equal-magnitude gradients by construction, for the ideal AdaDPO

*Correspondence to: qmao@inceptlabs.ai

Accepted at ICML 2026 Workshop on Decision-Making from Offline Datasets to Online Adaptation: Black-Box Optimization to Reinforcement Learning.

loss, which our Stable AdaDPO implementation approximates in practice. The principle requires only a few lines of code, introduces no sensitive hyperparameters beyond DPO's β, and extends to other pairwise contrastive preference losses. In terms of **contributions**, we (i) derive the gradient-balance condition for DPO-style losses, and prove that AdaDPO satisfies it exactly via stop-gradient operators (Section 3); (ii) show in preliminary experiments on Llama-3-8B-Instruct + UltraFeedback that AdaDPO outperforms DPO on AlpacaEval 2 in 81% of hyperparameter combinations, with an enlarged LC-over-WR margin in 88% of combinations, indicating effective mitigation of length bias (Section 5); and (iii) show that the self-adaptive principle generalizes to other pairwise contrastive preference losses. A schematic illustration of the DPO/AdaDPO contrast appears in Appendix A.1.

2 Background

Direct Preference Optimization. We are given a preference dataset $\mathcal{D} = \{(x, y_w, y_l)\}$ of triples consisting of a prompt x, a preferred response y_w, and a dispreferred response y_l, together with a reference policy π_{ref}. DPO [1] optimizes the policy π_θ by minimizing

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right], \quad (1)$$

where σ is the sigmoid function and $\beta > 0$ controls the deviation from π_{ref}.

The gradient imbalance pathology. Feng et al. [5] define $P_w = \pi_\theta(y_w \mid x)$, $P_l = \pi_\theta(y_l \mid x)$, $R_w = \pi_{\text{ref}}(y_w \mid x)$, $R_l = \pi_{\text{ref}}(y_l \mid x)$, and let $x_w = P_w/R_w$, $x_l = P_l/R_l$. The partial derivatives yield the gradient-magnitude ratios

$$\text{Ratio}_x = \left|\frac{\partial \mathcal{L}_{\text{DPO}}/\partial x_w}{\partial \mathcal{L}_{\text{DPO}}/\partial x_l}\right| = \frac{x_l}{x_w}, \qquad \text{Ratio}_P = \left|\frac{\partial \mathcal{L}_{\text{DPO}}/\partial P_w}{\partial \mathcal{L}_{\text{DPO}}/\partial P_l}\right| = \frac{P_l}{P_w}. \quad (2)$$

As training progresses and the policy learns to prefer y_w, both ratios fall below 1, meaning DPO suppresses y_l more aggressively than it promotes y_w [5].

Concurrent work on gradient imbalance. Two concurrent works also target DPO's gradient asymmetry. Ma et al. [11] (Balanced-DPO) introduce a global reweighting factor that balances average gradient contributions across the batch. Zhu et al. [12] (SGDPO) add an auxiliary *pilot* term that resamples a sub-sequence from the policy logits. AdaDPO differs by enforcing $|\partial\mathcal{L}/\partial P_w| = |\partial\mathcal{L}/\partial P_l|$ *exactly and per preference pair* via the closed-form ratio of Proposition 1, with no auxiliary terms, resampling, or extra forward passes, and no new tunable hyperparameters.

3 AdaDPO: Self-Adaptive Direct Preference Optimization

Adaptive Coefficients for Balanced Gradients. DPO's implicit reward margin uses the same coefficient β for both log-ratio terms [1]. We replace it with per-preference-pair coefficients β_w and β_l:

$$\Delta_{\text{AdaDPO}} = \beta_w \log \frac{P_w}{R_w} - \beta_l \log \frac{P_l}{R_l} = r(x, y_w) - r(x, y_l). \quad (3)$$

Proposition 1 (Gradient balance) *Let $\mathcal{L}$ be a DPO-style loss with implicit margin Δ_{AdaDPO} of the form* (3)*. Requiring $|\partial\mathcal{L}/\partial P_w| = |\partial\mathcal{L}/\partial P_l|$ yields the policy-space constraint*

$$\frac{\beta_w}{\beta_l} = \frac{P_w}{P_l}, \quad (4)$$

yielding strictly equal per-pair gradient magnitudes on the policy probabilities. (The ratio-space form $\beta_w/\beta_l = P_w R_l/(P_l R_w)$ instead balances the gradients on the importance ratios of Eq. (2)*; the two coincide when $R_w = R_l$. Our implementation uses the ratio-space form; see Appendix A.6.)*

Stop-Gradient Parameterization. The ratio in (4) depends on P_w and P_l, themselves outputs of the policy model. Setting β_w directly to these values would propagate gradients through the coefficients and break the intended balance. We treat the coefficients as constants during backpropagation using the stop-gradient operator $\mathrm{sg}(\cdot)$. Setting $\beta_l = \beta$ to preserve DPO's hyperparameter:

$$\beta_w = \beta \cdot \mathrm{sg}\left(\frac{P_w}{P_l}\right). \tag{5}$$

AdaDPO uses the same β as DPO and adds only a numerical clipping constant C on the adaptive ratio for stability. In our experiments we use the ratio-space variant $\beta_w/\beta_l = P_w R_l/(P_l R_w)$, with length normalization (Listing 1); the policy-space variant yields essentially identical performance. The full AdaDPO objective replaces the fixed β in DPO's loss with the adaptive coefficients:

$$\mathcal{L}_{\mathrm{AdaDPO}}(\pi_\theta; \pi_{\mathrm{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta_w \log\frac{P_w}{R_w} - \beta_l \log\frac{P_l}{R_l}\right)\right]. \tag{6}$$

Intuition and verification. The gradient on P_w is proportional to β_w/P_w. By setting $\beta_w \propto P_w$, we cancel the inverse-probability scaling that shrinks this gradient as the model becomes confident, preventing the "vanishing promotion" problem in vanilla DPO where updates stagnate once the model identifies y_w. Direct substitution into Proposition 1 verifies the balance for this ideal form: $|\partial\mathcal{L}_{\mathrm{AdaDPO}}/\partial P_w|/|\partial\mathcal{L}_{\mathrm{AdaDPO}}/\partial P_l| = (\beta_w/\beta_l)\cdot(P_l/P_w) = 1$.

Implementation. AdaDPO requires only a few lines of code on top of a standard DPO loss. We compute the adaptive ratio in log-space (avoiding numerical underflow), clip it at a fixed ceiling $C = 2$ for stability, and use the result to scale β_w. The modification relative to standard DPO is only the four lines computing `beta_w` as shown in the full implementation (Listing 1 in Appendix A.3). Because of length normalization and clipping, this implemented *Stable AdaDPO* only approximates the exact balance of Proposition 1: for $C \geq 1$ each pair's gradient ratio moves monotonically from DPO's P_l/P_w toward 1 but is capped at C, a bounded deviation we quantify in Appendix A.6.

4 Experimental Setup

(A) Base model and training data. We use Llama-3-8B-Instruct [13] as the base SFT model and train AdaDPO and DPO on the UltraFeedback preference dataset [14]. **(B) Hyperparameters and grid search.** We follow SimPO's training settings: batch size 128, maximum sequence length 2048, cosine learning rate schedule with 10% warmup, training for 1 epoch. We grid-search learning rate $\mathrm{lr} \in \{3, 5, 6, 10\} \times 10^{-7}$ and $\beta \in \{0.005, 0.01, 0.05, 0.1\}$, yielding 16 (lr, β) combinations per method (32 training runs total). We report both the best cell and aggregate statistics across the full grid. **(C) Evaluation benchmarks.** We evaluate on 3 standard instruction-following benchmarks with GPT-4 Turbo as the judge. ***AlpacaEval 2*** [15] consists of 805 queries and reports length-controlled win rate (LC), raw win rate (WR), and average response length. ***Arena-Hard v0.1*** [16] contains 500 challenging technical queries and reports WR with 95% confidence intervals. ***MT-Bench*** [17] covers 8 categories with 80 multi-turn questions on a 1–10 scale.

5 Results

Best Performance Comparison. Table 1 reports the strongest configuration per method across the 16 (lr, β) combinations. AdaDPO achieves the global best LC (48.3%) and WR (46.1%) on AlpacaEval 2, with shorter average response length than DPO on that benchmark. On Arena-Hard and MT-Bench, the two methods perform comparably with overlapping confidence intervals; we do not interpret these benchmarks as differentiating the methods (Table 1; 95% CIs in Appendix A.7).

Hyperparameter Sweep on AlpacaEval 2. Figure 1 shows AlpacaEval 2 performance for each of the 16 (lr, β) combinations. Aggregating across the 16 combinations: **LC**: AdaDPO outperforms DPO in 13/16 (81%) combinations; **WR**: AdaDPO outperforms DPO in 9/16 (56%) combinations; $\Delta = \mathbf{LC} - \mathbf{WR}$: AdaDPO produces a larger LC-over-WR margin in 14/16 (88%) combinations, indicating reduced length exploitation.

Table 1: Best performance per method across the 16 (lr, β) configurations. AdaDPO achieves the global best LC and WR on AlpacaEval 2 with the shortest average length on that benchmark.

	AlpacaEval 2				Arena-Hard				MT-Bench
Method	LC (%)	WR (%)	STD	Length	WR (%)	95%-Hi	95%-Lo	Length	Score
SFT	28.5	28.8	1.6	1976	26.3	28.1	24.7	589	7.58
DPO	46.4	44.5	1.8	1933	**42.9**	44.6	41.1	525	8.01
AdaDPO	**48.3**	**46.1**	1.8	1908	41.5	43.1	39.7	530	**8.03**

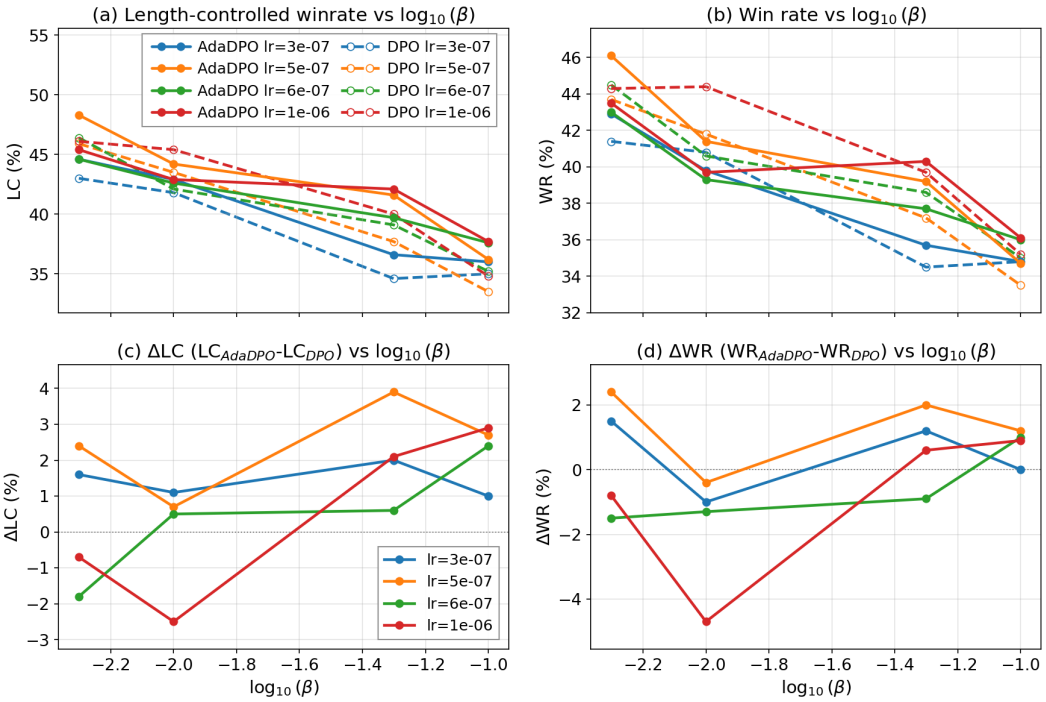

Figure 1: AlpacaEval 2 performance vs. $\log_{10}(\beta)$. (a) LC; (b) WR; (c) $\Delta\mathrm{LC} = \mathrm{LC}_{\mathrm{AdaDPO}} - \mathrm{LC}_{\mathrm{DPO}}$; (d) ΔWR. Solid: AdaDPO; dashed: DPO.

6 Discussion, Limitations, Conclusion

AdaDPO corrects DPO's structural gradient imbalance with per-pair adaptive coefficients that balance gradient magnitudes on P_w and P_l, with minimal code changes and no hyperparameters beyond DPO's β. The consistency of AdaDPO's gains across the 16-cell hyperparameter grid (LC win in 81% of combinations and an enlarged LC-over-WR margin in 88%) suggests that gradient balance, rather than careful β tuning, is the binding constraint on length-controlled performance for DPO-style objectives. The LC-over-WR margin improvement systematically tracks reduced response lengths, providing evidence that DPO's gradient asymmetry is mechanistically linked to length exploitation. Our experiments are limited to Llama-3-8B-Instruct on UltraFeedback; cross-model validation, transfer to alternative preference datasets; empirical evaluation of self-adaptive variants of SimPO, R-DPO, IPO, CPO, and ORPO are natural next steps. As an offline correction that strengthens the static-data learning stage of preference-based pipelines, AdaDPO suggests that the per-pair gradient-balance condition is a useful design lens for offline decision-making from preference data.

Acknowledgments and Disclosure of Funding

The authors would like to thank Lei Li and Nazhou Liu for their insightful suggestions, as well as Chenghua Wang, Zhifeng Li, Wenhui Chen, Jie Zhou, Xinmiao Yu, and Yanlin Fei for their valuable feedback and discussions.

References

[1] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[2] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[3] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

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

[5] Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. Towards analyzing and understanding the limitations of DPO: A theoretical perspective. *arXiv preprint arXiv:2404.04626*, 2024.

[6] Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024.

[7] Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization. *arXiv preprint arXiv:2403.19159*, 2024.

[8] Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*, 2023.

[9] Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. Contrastive preference optimization: Pushing the boundaries of LLM performance in machine translation. *arXiv preprint arXiv:2401.08417*, 2024.

[10] Jiwoo Hong, Noah Lee, and James Thorne. ORPO: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691*, 2024.

[11] Qinwei Ma, Jingzhe Shi, Can Jin, Jenq-Neng Hwang, Serge Belongie, and Lei Li. Gradient imbalance in direct preference optimization. *arXiv preprint arXiv:2502.20847*, 2025.

[12] Wenqiao Zhu, Ji Liu, Lulu Wang, Jun Wu, and Yulun Zhang. SGDPO: Self-guided direct preference optimization for language model alignment. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 12366–12383, Vienna, Austria, 2025. Association for Computational Linguistics.

[13] AI@Meta. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[14] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. UltraFeedback: Boosting language models with high-quality feedback. In *International Conference on Machine Learning (ICML)*, 2024.

[15] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled AlpacaEval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

[16] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From live data to high-quality benchmarks: The Arena-Hard pipeline. LMSYS Blog, 2024. `https://lmsys.org/blog/2024-04-19-arena-hard/`.

[17] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023.

A Technical appendices and supplementary material

A.1 Schematic Comparison of DPO and AdaDPO

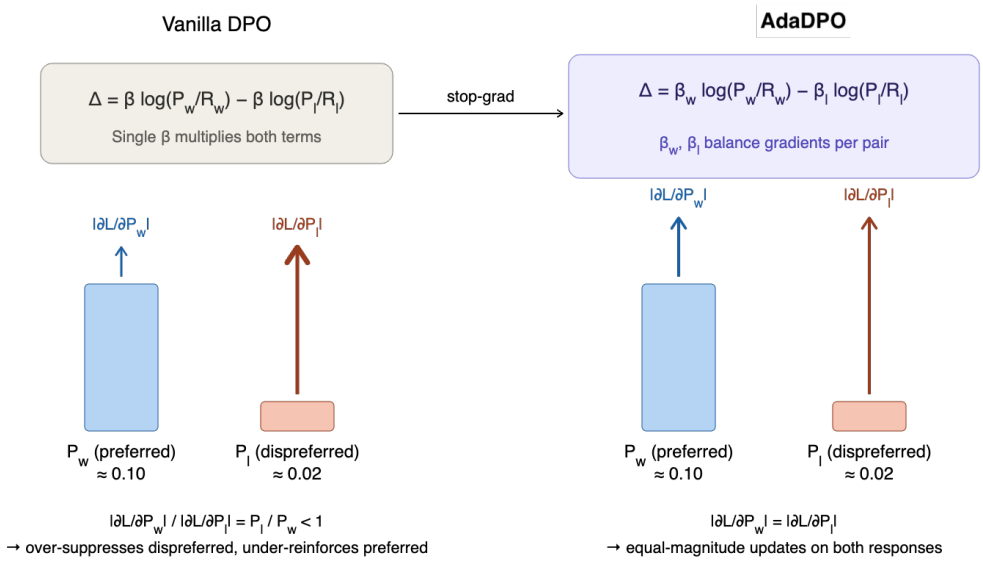

Figure 2: Schematic comparison of vanilla DPO (left) and AdaDPO (right). DPO's shared coefficient β produces systematically larger gradients on the dispreferred response; AdaDPO's per-pair coefficients are designed to yield equal-magnitude updates on P_w and P_l.

A.2 Optimization Dynamics

Figure 3 compares AdaDPO and DPO under two representative β values (0.05 and 0.01) at fixed learning rate 5×10^{-7}. AdaDPO consistently achieves lower evaluation loss (Figure 3a), higher reward accuracy (Figure 3b), larger $\beta \times \mathrm{KL}$ divergence margin (Figure 3c), and larger reward margin $r(x, y_w) - r(x, y_l)$ (Figure 3d). The performance ordering across all metrics consistently places AdaDPO above DPO at matched β. While the $\beta \times \mathrm{KL}$ margin is dominated by the KL constraint strength, the other metrics—more directly tied to alignment quality—benefit clearly from balanced gradient updates, indicating that a large KL margin alone is insufficient: balanced updates are key to translating margin into improved performance.

A.3 Implementation Details

The complete PyTorch implementation appears in Listing 1.

Listing 1: PyTorch implementation of AdaDPO with the length-normalized adaptive ratio.

```
def adadpo_loss(pi_logps, ref_logps, yw_idx, yl_idx, beta,
                C=2.0, yw_lens=None, yl_lens=None):
    pi_yw, pi_yl = pi_logps[yw_idx], pi_logps[yl_idx]
    ref_yw, ref_yl = ref_logps[yw_idx], ref_logps[yl_idx]

    # Adaptive coefficient via stop-gradient (length-normalized)
    lw = yw_lens.float().detach()
    ll = yl_lens.float().detach()
    log_ratio = (pi_yw.detach() / lw - pi_yl.detach() / ll
                 - ref_yw.detach() / lw + ref_yl.detach() / ll)
    beta_w = beta * torch.clamp(log_ratio.exp(), max=C)

    # Standard DPO-style loss with adaptive beta_w
    delta = beta_w * (pi_yw - ref_yw) - beta * (pi_yl - ref_yl)
    return -F.logsigmoid(delta)
```

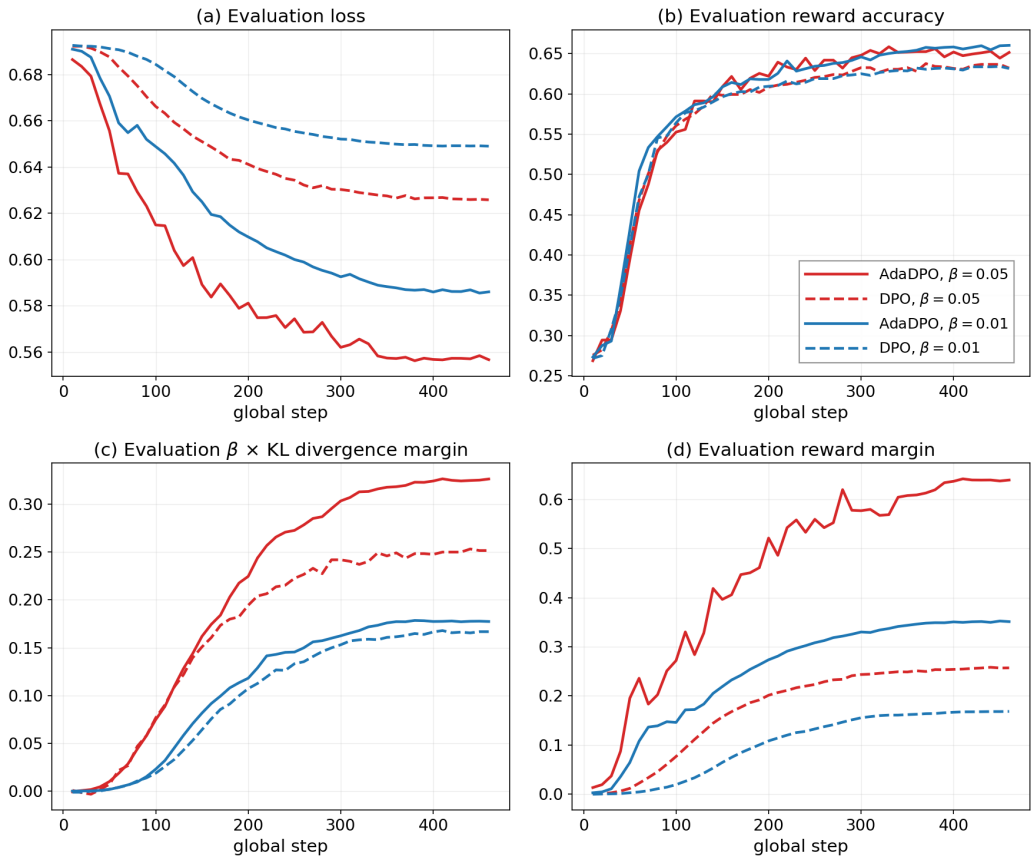

Figure 3: Training-dynamics comparison on the validation set. (a) Evaluation loss; (b) reward accuracy ($\Pr[r(x, y_w) > r(x, y_l)]$); (c) $\beta \times$ KL divergence margin; (d) reward margin $r(x, y_w) - r(x, y_l)$. Solid: AdaDPO, dashed: DPO; red: $\beta = 0.05$, blue: $\beta = 0.01$. AdaDPO's loss and reward margin are scaled by the adaptive coefficient β_w, so absolute values are not directly comparable to DPO's; reward accuracy (b) is scale-invariant.

The length-normalized variant shown is used for all main-text experiments; this corresponds to setting exponents $\alpha_w = 1/|y_w|$, $\alpha_l = 1/|y_l|$ in the exponent generalization

$$\beta_w = \beta \cdot \mathrm{sg}\left(\frac{(P_w/R_w)^{\alpha_w}}{(P_l/R_l)^{\alpha_l}} \right), \tag{7}$$

which recovers DPO when $\alpha_w = \alpha_l = 0$ and the base unnormalized form of Equation (5) when $\alpha_w = \alpha_l = 1$. The unnormalized variant produces less stable training dynamics on long responses but achieves comparable best-case performance after appropriate hyperparameter tuning. The full grid-search artifacts (training logs, evaluation outputs, and per-configuration metrics for all 32 runs) will be released with the camera-ready version.

Computational Resources. Each AdaDPO and DPO training run used a Llama-3-8B-Instruct base model with a batch size of 128 and a sequence length of 2048 for one epoch on UltraFeedback [14]. The average training runtime for each of the 32 runs was 2.9 hours (10,416s) conducted on $8 \times H100$ GPU nodes using bf16 precision. For the evaluation on AlpacaEval 2, Arena-Hard, and MT-Bench, we utilized either $8 \times H100$ or $8 \times A100$ GPUs, depending on cluster availability. We note that evaluation times vary based on the specific inference pipelines used.

A.4 Ceiling Constant Ablation

We ablate the ceiling constant C over $\{1.5, 2, 2.5, 3, 4, 5, 10\}$ at $\beta = 0.01$, taking the best LC across two learning rates (5×10^{-7} and 1×10^{-6}) per value of C. The optimum is $C = 2$ (LC 44.2%, WR 42.7%); performance is robust on $[1.5, 2.5]$ and degrades beyond $C \geq 5$.

Table 2: Effect of ceiling constant C on AlpacaEval 2 performance ($\beta = 0.01$).

C	LC (%)	WR (%)	Length
1.5	43.7	37.3	1772
2.0	**44.2**	**42.7**	1941
2.5	40.5	37.5	1867
3.0	41.6	37.8	1838
4.0	39.8	37.1	1881
5.0	38.4	36.0	1886
10.0	34.9	34.8	1979

A.5 Generalization to DPO Variants: Derivations

The AdaDPO principle—replacing fixed coefficients with self-adaptive stop-gradient-based coefficients—applies to any pairwise contrastive loss of the form

$$\mathcal{L} = f\left(\beta_w \log \frac{P_w}{R_w} - \beta_l \log \frac{P_l}{R_l} + \lambda\right), \tag{8}$$

where f is any differentiable function (e.g., $f(x) = \log \sigma(x)$ for DPO, SimPO, R-DPO, CPO; $f(x) = x^2$ for IPO). For any such loss, requiring $|\partial\mathcal{L}/\partial P_w| = |\partial\mathcal{L}/\partial P_l|$ yields $\beta_w/\beta_l = P_w/P_l$ (the ratio-space counterpart $\beta_w/\beta_l = P_w R_l/(P_l R_w)$ enforces the analogous balance on the importance ratios P/R, coinciding when $R_w = R_l$). For ORPO, whose loss involves $\log[P/(1-P)]$, the balance condition gives $\beta_w/\beta_l = \mathrm{sg}(P_w(1-P_w)/[P_l(1-P_l)])$.

Beyond symmetric balancing ($k = 1$), the same framework allows an optional scaling factor k that biases the update toward reinforcing preferred responses ($k > 1$) or suppressing dispreferred responses ($k < 1$). Concretely, one can set

$$\frac{\beta_w}{\beta_l} = k \cdot \mathrm{sg}\left(\frac{P_w}{P_l}\right),$$

leaving all other aspects of the loss unchanged. In this paper we always fix $k = 1$ and report results only for the balanced setting, which we find to be the most robust across general benchmarks. The exponent generalization framework introduced in Equation (7) extends naturally to all variants in Table 3 by replacing each adaptive coefficient with its (α_w, α_l)-parameterized counterpart.

Table 3: Self-adaptive counterparts of preference-optimization objectives under the AdaDPO principle. Each method preserves its original hyperparameters (DPO, CPO, R-DPO, ORPO, and SimPO use β; IPO has no β, hence $\beta_l = 1$).

Method	Self-adaptive objective	Adaptive coefficients
DPO	$\log \sigma\big(\beta_w \log \frac{P_w}{R_w} - \beta_l \log \frac{P_l}{R_l}\big)$	$\beta_l = \beta,\ \ \beta_w = \beta \cdot \mathrm{sg}\big(\frac{P_w R_l}{P_l R_w}\big)$
IPO	$\big(\beta_w \log \frac{P_w}{R_w} - \beta_l \log \frac{P_l}{R_l} - \frac{1}{2\tau}\big)^2$	$\beta_l = 1,\ \ \beta_w = \mathrm{sg}\big(\frac{P_w R_l}{P_l R_w}\big)$
R-DPO	$\log \sigma\big(\beta_w \log \frac{P_w}{R_w} - \beta_l \log \frac{P_l}{R_l} - \alpha(\lvert y_w\rvert - \lvert y_l\rvert)\big)$	$\beta_l = \beta,\ \ \beta_w = \beta \cdot \mathrm{sg}\big(\frac{P_w R_l}{P_l R_w}\big)$
CPO	$\log \sigma\big(\beta_w \log P_w - \beta_l \log P_l\big)$	$\beta_l = \beta,\ \ \beta_w = \beta \cdot \mathrm{sg}\big(\frac{P_w}{P_l}\big)$
ORPO	$\log P_w + \log \sigma\big(\beta_w \log \frac{P_w}{1-P_w} - \beta_l \log \frac{P_l}{1-P_l}\big)$	$\beta_l = \beta,\ \ \beta_w = \beta \cdot \mathrm{sg}\big(\frac{P_w(1-P_w)}{P_l(1-P_l)}\big)$
SimPO	$\log \sigma\big(\frac{\beta_w}{\lvert y_w\rvert} \log P_w - \frac{\beta_l}{\lvert y_l\rvert} \log P_l - \gamma\big)$	$\beta_l = \beta,\ \ \beta_w = \beta \cdot \mathrm{sg}\big(\frac{\lvert y_w\rvert P_w}{\lvert y_l\rvert P_l}\big)$

Ablation of Balancing Space. While our main experiments utilize the ratio-space variant $\beta_w/\beta_l = P_w R_l/(P_l R_w)$ as shown in Listing 1, we conducted an internal ablation using the simpler policy-space variant $\beta_w/\beta_l = P_w/P_l$. We observed essentially identical AlpacaEval 2 performance across the (lr, β) grid for both variants. This confirms that the self-adaptive principle is robust to the specific probability space in which gradient equality is enforced, provided the stop-gradient operator is applied correctly.

A.6 Stable AdaDPO: Additional Details

The exponent family introduced in Equation (7) accommodates several special cases beyond DPO ($\alpha = 0$), base AdaDPO ($\alpha = 1$), and Stable AdaDPO ($\alpha = 1/|y|$). Intermediate values of α smoothly interpolate between these regimes. We empirically observed that Stable AdaDPO yields the most stable training across the broadest range of β values; hence its use throughout Section 5. Asymmetric exponents ($\alpha_w \neq \alpha_l$) are an interesting direction for future work, particularly for settings where preferred and dispreferred responses differ systematically in length or complexity (e.g., reasoning-style preferences with verbose chain-of-thought y_w and terse incorrect y_l).

Ideal vs. implemented AdaDPO. For clarity, we distinguish between the *ideal* AdaDPO construction and the *Stable AdaDPO* variant used in all reported experiments. The ideal objective enforces per-pair equality of gradient magnitudes on P_w and P_l exactly via the stop-gradient parameterization of Proposition 1, in the absence of any additional normalization or clipping. In practice, our implementation introduces length normalization (via the exponent family in Eq. (7)) and a ceiling constant C on the adaptive ratio (Listing 1) to improve numerical stability and robustness on long responses. This *Stable AdaDPO* instantiation preserves the self-adaptive, per-pair nature of the coefficients while approximating the ideal gradient-balance condition under these pragmatic constraints. All experimental results in Section 5 use the Stable AdaDPO variant.

Magnitude of the deviation. Recall from Eq. (2) that DPO's per-pair gradient-magnitude ratio is P_l/P_w, which falls below 1 as the policy comes to prefer y_w; the ideal balanced coefficient raises this ratio to exactly 1, while Stable AdaDPO multiplies it by the clipped adaptive factor $\min(\rho, C)$ with $\rho = P_w/P_l$ for the policy-space form. Hence every pair whose preferred response is at most C times as probable as its dispreferred one ($P_w/P_l \leq C$) attains exact balance, and only beyond this point does the ceiling bind: there the realized ratio is exactly C times DPO's, so Stable AdaDPO remains a factor-C improvement toward balance however confident the model becomes. For $C = 2$, a pair at $P_w/P_l = 5$ has a DPO ratio of 0.2 versus 0.4 under Stable AdaDPO, and at $P_w/P_l = 10$, 0.1 versus 0.2.

A.7 Additional Benchmark Results

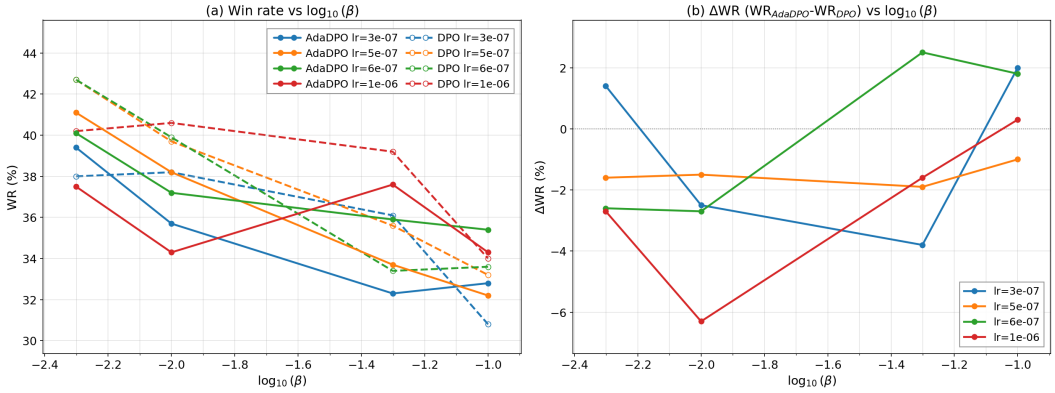

Figure 4: Arena-Hard performance comparison between AdaDPO and DPO. (a) Win rate vs. $\log_{10}(\beta)$; (b) ΔWR. The 95% confidence intervals on the best configurations overlap, indicating no statistically significant difference between methods on this length-uncontrolled benchmark.

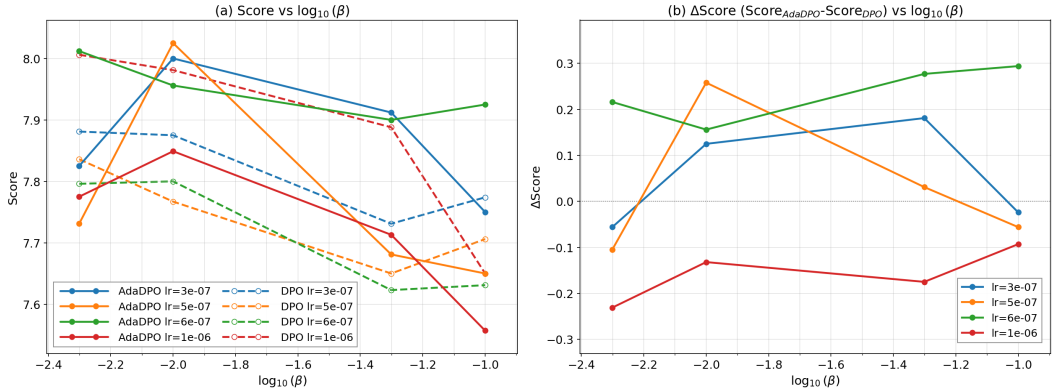

Figure 5: MT-Bench performance comparison between AdaDPO and DPO. The 0.02-point gap between AdaDPO (8.03) and DPO (8.01) is below the resolution of the 1–10 scoring scale and 80-question evaluation set.

Broader Impact

AdaDPO improves the alignment of large language models with human preferences, with potential positive impacts on the safety, helpfulness, and reliability of deployed LLMs. As a methodological correction that is agnostic to the underlying preference data, its specific societal impact follows directly from the choice of the preference dataset; the method does not introduce new risks beyond those already present in standard preference optimization pipelines.