PRECONDITIONED DIRECT PREFERENCE OPTIMIZATION: GEOMETRY AND ACCELERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning from human feedback (RLHF) has become a central tool for aligning large language models with human preferences. Direct preference optimization (DPO) recasts RLHF as a policy optimization problem without explicitly estimating the reward function, and enhances the stability and efficiency of the training pipeline. Existing methods for DPO update based on policy gradients, and only the linear convergence rate is guaranteed. Therefore, to achieve acceleration, we consider using second-order information. A natural candidate is the Newton direction, which, however, involves the product of a Hessian inverse and the negative gradient, appearing computationally intractable in large-scale optimization. To circumvent this, we exploit the structure of DPO and, remarkably, characterize the regularized Newton direction via fully first-order information. This direction—which is used to propose a preconditioned method for DPO (PDPO)—encodes the geometry of the *comparison graph*, where each response is a node and each pairwise preference induces an edge. Consequently, we prove that updating along the Newton direction achieves quadratic convergence, and the updates along preconditioned directions restore the superlinear or the quadratic convergence rate under mild assumptions. Numerical experiments demonstrate that the proposed method achieves superior convergence rate and accuracy than the baselines.

1 INTRODUCTION

Alignment with human preferences has achieved great success in the training of large language models (LLMs), leading to responses of higher quality (Ouyang et al., 2022). Reinforcement learning from human feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022) serves as the typical paradigm. Specifically, the LLM generates several responses to each prompt, and human annotators compare the responses in pairs and indicate which response they prefer. A reward model is then fitted to these pairwise preferences, and the LLM is optimized based on the learned reward by a reinforcement learning algorithm, e.g., the proximal policy optimization (PPO) (Schulman et al., 2017), while a Kullback–Leibler (KL) penalty keeps the LLM close to the supervised fine-tuned model (Ouyang et al., 2022).

Despite the effectiveness, the two-stage procedure of RLHF can incur training instability and computational overhead (Casper et al., 2023), and thus direct preference optimization (DPO) (Rafailov et al., 2023) has emerged as an efficient alternative. Noting that the KL-penalized reward maximization of RLHF admits a closed-form optimal policy, DPO reparameterizes the reward through the difference of log-probabilities predicted by the trained policy and the reference policy, and hence obtains the *implicit reward* of each response, thereby avoiding fitting a reward model explicitly. As a result, DPO has garnered growing interest: Xu et al. (2024) compared it with PPO-based RLHF, and a line of works has extended the loss function, e.g., to curb overfitting to the preferences (Azar et al., 2024), to benefit from rankings of several responses (Liu et al., 2025), and to learn from a binary signal of whether a single response is desirable (Ethayarajh et al., 2024).

Through the lens of convergence rate, analysis on methods for DPO has also attracted wide attention. Specifically, Chandran et al. (2025) proved that policy gradient descent on the DPO loss converges linearly. More relevantly, two works targeted the acceleration of DPO. He et al. (2024) viewed the update method based on responses generated by the latest policy, as a proximal point method and accelerated it by Nesterov's extrapolation. Shi et al. (2025) designed new samplers of the compared

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

pairs to make DPO converge quadratically, which requires every pair of responses to be compared, i.e., the comparison graph is complete. Consequently, noting that DPO is essentially an optimization problem, we are motivated to take into account second-order methods to enhance the efficiency.

To this end, a natural candidate is the Newton method, which, however, involves the product of a Hessian inverse and the gradient, which appears computationally intractable in large-scale optimization. To circumvent this difficulty, we are inspired by Yang et al. (2025), which studied the RLHF problem, and unveiled that the Hessian-inverse-vector product can be characterized by sampling fully first-order information. In the same spirit, this work exploits the second-order structure of DPO, relating the (regularized) Newton direction with a random walk over the comparison graph, which guides the development of the efficient algorithm for DPO.

Contributions. The main contributions are summarized as follows.

Firstly, we prove that, on a connected comparison graph, the Hessian—denoted by $\nabla^2\mathcal{L}$—of the DPO loss is singular along a one-dimensional subspace. Nevertheless, we exploit the invariance of the problem to recast it on a quotient space, on which the quotient Hessian is proved to be invertible. This property yields a well-defined quotient Newton step, which can be lifted to the original space, producing the step along the pseudoinverse of $\nabla^2\mathcal{L}$ applied to the negative gradient, i.e., $u^+ = u - (\nabla^2\mathcal{L}(u))^\dagger\nabla\mathcal{L}(u)$; see Proposition 4.3.

Secondly, to enhance the training stability, we introduce a regularized term with $\gamma > 0$ to give the *preconditioned direction* $(\nabla^2\mathcal{L}(u) + \gamma I)^{-1}\nabla\mathcal{L}(u)$, which tends to the Newton direction as $\gamma \to 0$. More importantly, we find that the preconditioned direction—which includes the Hessian operator—can be characterized by an expectation of first-order information over random walks on the comparison graph. As a result, when the preference dataset is given a priori, an unbiased estimator of the preconditioned direction is constructed in Theorem 5.1; as an extension, Proposition 5.2 adapts the theory to the online scenario.

Finally, we propose a preconditioned method for DPO, abbreviated as PDPO. Moreover, we establish a locally quadratic convergence rate of the Newton step under connected comparison graphs (Theorem 6.1); furthermore, the preconditioned updates can also retain a superlinear or quadratic convergence under appropriate schedules of γ; see Proposition 6.2. In addition, the efficiency of PDPO is validated by a synthetic problem and two language model tasks.

2 Related Work

We review some developments in DPO and preconditioning methods for general problems.

Direct preference optimization. DPO (Rafailov et al., 2023) adopted the policy parameters to represent the implicit reward, which is fit to pairwise preferences through a classification loss, and a line of works has modified the loss function, for instance IPO (Azar et al., 2024), KTO (Ethayarajh et al., 2024), ORPO (Hong et al., 2024) and SimPO (Meng et al., 2024). In addition, several works have learned from a ranking of several responses to each prompt. RRHF (Yuan et al., 2023) trained the model so that its length-normalized log-probabilities of the responses follow the ranking, PRO (Song et al., 2024) replaced the Bradley–Terry likelihood of a pair by a Plackett–Luce-type likelihood according to the whole ranking, and LiPO (Liu et al., 2025) applied the DPO loss to every pair in the ranking with rank-dependent weights.

The convergence analysis of methods for DPO has also gained momentum in recent years. He et al. (2024) viewed DPO as a proximal point method and accelerated it by Nesterov-type extrapolation; Shi et al. (2025) showed, for the policy gradient method, that uniform sampling of the compared pair renders a linear convergence rate, and that the developed sampler DPO-Mix-P, which reweights the probabilities of responses, leads to a quadratic convergence rate; and Chandran et al. (2025) bounded the number of gradient-descent iterations on DPO with the condition number of the comparison-graph Laplacian, and used the bound to select preference data. Kim et al. (2026) proved exponential convergence of on-policy DPO for linear softmax policies under large batches.

Newton methods and preconditioning. Preconditioning rescales a gradient direction by multiplying it with a matrix, aiming to reduce the condition number and to accelerate convergence (Benzi,

2002; Saad, 2003). From this perspective, the Newton method applies the inverse Hessian to the gradient and converges locally quadratically near a nondegenerate minimizer (Nocedal & Wright, 2006). Liu & Nocedal (1989) studied the limited-memory BFGS method, which approximates the inverse Hessian using a small number of previous steps and gradient differences. In statistical learning, Amari (1998) used the Fisher information matrix to precondition the gradient, and Martens & Grosse (2015) developed Kronecker-factored approximations of this matrix for neural-network training. Preconditioning techniques have also been developed by exploiting structures of the specific problems: Mishra & Sepulchre (2016) constructed preconditioners from the Hessian of the Lagrangian for constrained and quotient problems, while Gao et al. (2025) designed block-diagonal preconditioners for optimization problems on product spaces.

3 Preliminaries: RLHF and DPO

Let x denote a prompt, and let $\mathcal{Y}$ be a finite set of responses. A policy π associates to each prompt x a probability distribution $\pi(\cdot \mid x)$ over $\mathcal{Y}$. The reference policy π_{ref} is the model from which alignment starts, typically obtained by supervised fine-tuning. Given a pair of responses conditioned on the same prompt, a human annotator expresses a preference for one of them. We denote by y_w the preferred response and by y_l the dispreferred one, write the preference as $y_w \succ y_l$, and call y_w and y_l the winner and the loser, respectively. Let $\sigma(t) = 1/(1+\mathrm{e}^{-t})$ denote the logistic function. The Bradley–Terry model (Bradley & Terry, 1952) assumes that preferences are generated by a latent reward r, that is,

$$p(y_w \succ y_l \mid x) \;=\; \sigma\big(r(x, y_w) - r(x, y_l)\big). \tag{1}$$

RLHF (Christiano et al., 2017; Ouyang et al., 2022) firstly fits a reward model r by minimizing $-\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}[\log \sigma(r(x,y_w) - r(x,y_l))]$, and then trains the policy to maximize

$$\mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi(\cdot|x)}[r(x,y)] - \beta\, D_{\mathrm{KL}}[\pi(\cdot \mid x) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid x)], \tag{2}$$

where D_{KL} is the KL-divergence between two distributions, and the regularization coefficient $\beta > 0$ controls the balance between maintaining the original reference model π_{ref} and incorporating new preferences. In fact, the objective (2) admits the closed-form maximizer: $\pi(y \mid x) = \pi_{\mathrm{ref}}(y \mid x)\exp(r(x,y)/\beta)/Z(x)$, with $Z(x) = \sum_{y\in\mathcal{Y}} \pi_{\mathrm{ref}}(y \mid x)\exp(r(x,y)/\beta)$ as the partition function. Therefore, we can reparameterize the reward function through the policies as follows,

$$r(x,y) \;=\; \beta \log \frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + \beta \log Z(x). \tag{3}$$

Parameterizing the policy with the weight θ and substituting (3) into (1), the DPO gives the following optimization problem,

$$\min_{\theta} \quad \mathcal{L}(\theta) \;=\; -\,\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\log \sigma\Big(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)}\Big)\Big], \tag{4}$$

which aims to increase the likelihood of the winning response and to decrease that of the losing response (Rafailov et al., 2023).

4 Geometry of comparison graph and Newton step of DPO

To simplify the discussion, we first fix a prompt x, and let $\mathcal{Y} = \{y_1, \ldots, y_K\}$ index the responses conditioned on x, which are compared to give the preference. Consider the variable $u \in \mathbb{R}^K$ with entries parameterized by

$$u_k := \log \frac{\pi_\theta(y_k \mid x)}{\pi_{\mathrm{ref}}(y_k \mid x)}, \quad \text{for} \quad y_k \in \mathcal{Y}.$$

Let $\mathcal{E}$ denote the set of ordered comparisons $e = (w, l)$, where response y_w is preferred to y_l, and let $n_e > 0$ be a fixed comparison weight. Empirical counts are one choice of these weights; positive real weights are also allowed. We set $n_e = 0$ for comparisons that do not appear in $\mathcal{E}$. The single-prompt DPO objective is the weighted sum

$$\mathcal{L}(u) = -\sum_{e=(w,l)\in\mathcal{E}} n_e \,\log \sigma(\beta(u_w - u_l))\,. \tag{5}$$

Let $\mathbf{e}_y$ denote the y-th standard basis vector of $\mathbb{R}^K$. Direct differentiation gives

$$\nabla\mathcal{L}(u) = -\beta \sum_{e=(w,l)\in\mathcal{E}} n_e \left[1 - \sigma(\beta(u_w - u_l))\right](\mathbf{e}_w - \mathbf{e}_l), \tag{6}$$

$$\nabla^2\mathcal{L}(u) = \beta^2 \sum_{e=(w,l)\in\mathcal{E}} n_e\, \sigma(\beta(u_w - u_l))\left[1 - \sigma(\beta(u_w - u_l))\right](\mathbf{e}_w - \mathbf{e}_l)(\mathbf{e}_w - \mathbf{e}_l)^\top. \tag{7}$$

Let $\mathcal{G} = (\mathcal{Y}, \mathcal{E})$ denote the underlying undirected comparison graph, ignoring the orientation of comparisons in $\mathcal{E}$: two responses y_w, y_l are joined whenever $n_{(w,l)} + n_{(l,w)} > 0$. In this view, the Hessian $\nabla^2\mathcal{L}$ turns out to be the weighted Laplacian of the graph $\mathcal{G}$, which implies that $\nabla^2\mathcal{L}$ is positive semidefinite and thus $\mathcal{L}(u)$ is convex. To characterize the positivity of the Hessian more precisely, we impose the following assumption.

Assumption 4.1. The comparison graph $\mathcal{G} = (\mathcal{Y}, \mathcal{E})$ is connected.

The connectivity reveals that any two responses are implicitly linked by a path of comparisons, under which we find that the Hessian vanishes exactly on the one-dimensional space $\operatorname{span}\{\mathbf{1}\}$, where we use $\mathbf{1}$ to denote the all-ones vector.

Proposition 4.2. *For every $u \in \mathbb{R}^K$ the Hessian (7) is positive semidefinite and satisfies $\nabla^2\mathcal{L}(u)\mathbf{1} = 0$. Under Assumption 4.1, we have $\ker \nabla^2\mathcal{L}(u) = \operatorname{span}\{\mathbf{1}\}$.*

Proof. See Section A.1. □

Extending the spirit of the above proposition, we note that the objective (5) takes the same value at u and at $u + c\mathbf{1}$ for any $c \in \mathbb{R}$, as it only depends on the difference $u_w - u_l$. Therefore, we can recast the problem onto the quotient space $\mathcal{M} := \mathbb{R}^K / \operatorname{span}\{\mathbf{1}\}$—of which the equivalence class is $[u] = u + \operatorname{span}\{\mathbf{1}\}$—that is,

$$\min_{[u]\in\mathcal{M}} \ \overline{\mathcal{L}}([u]) := \mathcal{L}(u). \tag{8}$$

The $\mathcal{M}$ is a quotient space of dimension $K - 1$, which admits a quotient map $\pi : \mathbb{R}^K \to \mathcal{M} : u \mapsto [u]$. Following the theory of quotient space (Boumal, 2023, §9) and denoting the differential of π by $\mathrm{D}\pi$, we introduce $\mathcal{H} := \{v \in \mathbb{R}^K : \mathbf{1}^\top v = 0\}$ and define the *horizontal lift* of any direction $\xi \in \mathcal{M}$ as the unique $\overline{\xi} \in \mathcal{H}$ such that $\mathrm{D}\pi(u)[\overline{\xi}] = \xi$. Consequently, we can represent points and directions on $\mathcal{M}$ by those on $\mathbb{R}^K$, as well as the gradient and Hessian of $\overline{\mathcal{L}}$ under the inner product induced from $\mathbb{R}^K$. Therefore, for the quotient problem (8), we prove in the following proposition that the Hessian on $\mathcal{M}$ is positive definite, and thus the quotient Newton step is well defined.

Proposition 4.3. *Under Assumption 4.1, given any $[u] \in \mathcal{M}$, the quotient Newton equation $\operatorname{Hess}\overline{\mathcal{L}}([u])[\xi] = -\operatorname{grad}\overline{\mathcal{L}}([u])$ has a unique solution ξ, and the horizontal lift of ξ is*

$$\overline{\xi} \ = \ -\big(\nabla^2\mathcal{L}(u)\big)^\dagger \nabla\mathcal{L}(u). \tag{9}$$

The update $[u] \mapsto [u + \overline{\xi}]$ corresponds to $u^+ = u - \big(\nabla^2\mathcal{L}(u)\big)^\dagger\nabla\mathcal{L}(u)$ in the space $\mathbb{R}^K$.

Proof. See Section A.2. □

Although $\nabla^2\mathcal{L}(u)$ is singular by Proposition 4.2, a Newton step becomes available once the problem is cast on the quotient space $\mathcal{M}$, which induces the update based on a pseudoinverse in $\mathbb{R}^K$; the locally quadratic convergence rate is analyzed in Section 6.

5 Regularized Newton step and first-order characterization

Proposition 4.3 sheds light on the second-order method for DPO, and to enhance the stability of the training, we take a parameter $\gamma > 0$ and consider the regularized Newton direction at u as follows,

$$d(u) \ = \ -\big(\nabla^2\mathcal{L}(u) + \gamma I\big)^{-1}\nabla\mathcal{L}(u), \tag{10}$$

also called the damped Newton direction or the *preconditioned DPO direction*. In addition, we note that $\nabla\mathcal{L}(u) \in \mathcal{H}$ and $\nabla^2\mathcal{L}(u)+\gamma I$ maps $\mathcal{H}$ onto $\mathcal{H}$. Therefore, $d(u)$ tends to the exact Newton step (9) as $\gamma \to 0$, and is dominated by the gradient direction $-\nabla\mathcal{L}(u)/\gamma$ for large γ.

A straightforward computation of $d(u)$ in (10) requires solving the linear system of size K, which costs $O(K^3)$ operations in general, affordable for scenarios where only few responses of one prompt are recorded. However, when K is large, or more generally, when the responses are generated in an online fashion, ranging over the output space of the language model, the matrix $\nabla^2\mathcal{L}$ cannot be formed explicitly. To circumvent this, we investigate the preconditioned direction involving a Hessian inverse, and find that it can be expressed through a random walk on the comparison graph and be estimated by sampling first-order information.

5.1 Sampling through a random walk over the comparison graph

For distinct responses y, z, write $n_{yz} := n_{(y,z)} + n_{(z,y)}$ for their total comparison weight. The Hessian formula (7) then gives

$$[\nabla^2\mathcal{L}(u)]_{yz} = \begin{cases} -\beta^2 n_{yz}\,\sigma\big(\beta(u_y-u_z)\big)\big[1-\sigma\big(\beta(u_y-u_z)\big)\big], & y \neq z, \\ -\sum_{z'\neq y}[\nabla^2\mathcal{L}(u)]_{yz'}, & y = z. \end{cases} \tag{11}$$

Since $n_{yz} \geq 0$ and $\sigma(t)[1-\sigma(t)] > 0$, each off-diagonal entry is nonpositive, thereby vanishing only when $n_{yz} = 0$. Each diagonal entry is a sum of nonnegative terms and is positive whenever the response has an incident edge of positive weight. Therefore, the Hessian is the weighted Laplacian of $\mathcal{G}$, which inspires us to define a random walk over the graph. Specifically, set $X_0 = y \in \mathcal{Y}$ as the start of the walk, and given X_k ($k = 0, 1, 2, \dots$) the walk either stops and outputs X_k, or moves to a next response $X_{k+1} = z \neq X_k$, with the respective probabilities

$$\frac{\gamma}{\gamma + [\nabla^2\mathcal{L}(u)]_{X_kX_k}} \qquad \text{and} \qquad \frac{-[\nabla^2\mathcal{L}(u)]_{X_kz}}{\gamma + [\nabla^2\mathcal{L}(u)]_{X_kX_k}}.$$

Note that the regularization parameter $\gamma > 0$ serves as the signal to stop the walk. Write $\mathsf{R}_\gamma(z \mid y)$ for the probability that the walk started at y stops at z, and R_γ for the $K \times K$ matrix with entries $[\mathsf{R}_\gamma]_{yz} = \mathsf{R}_\gamma(z \mid y)$, each row of which represents a probability distribution.

For a walk started at y and an endpoint z, summing the probability of stopping immediately (if $y = z$) and the probabilities of first moving to $y' \neq y$ and subsequently stopping at z gives the recursion,

$$\mathsf{R}_\gamma(z \mid y) \;=\; \frac{\gamma}{\gamma + [\nabla^2\mathcal{L}(u)]_{yy}}\,\mathbb{1}(y=z) \;+\; \sum_{y'\neq y} \frac{-[\nabla^2\mathcal{L}(u)]_{yy'}}{\gamma + [\nabla^2\mathcal{L}(u)]_{yy}}\,\mathsf{R}_\gamma(z \mid y'),$$

which, equivalently, can be rearranged into a compact matrix form, $(\gamma I + \nabla^2\mathcal{L}(u))\,\mathsf{R}_\gamma = \gamma I$, and thus we have

$$\mathsf{R}_\gamma \;=\; \gamma\big(\gamma I + \nabla^2\mathcal{L}(u)\big)^{-1}. \tag{12}$$

This equality plays a crucial role in the following theorem to realize $d(u)$ as an expectation of first-order information sampled through the random walk over $\mathcal{G} = (\mathcal{Y}, \mathcal{E})$. Define the total comparison weight $N = \sum_{e\in\mathcal{E}} n_e$ and the sampling distribution $\mu(e) = n_e/N$ on $\mathcal{E}$.

Theorem 5.1 (Random-walk representation of the regularized Newton direction)**.** *Let $\gamma > 0$ and write the regularized Newton direction as $d(u) = -\big(\gamma I + \nabla^2\mathcal{L}(u)\big)^{-1}\nabla\mathcal{L}(u)$. Then, we have*

$$d(u) \;=\; \frac{\beta N}{\gamma}\,\mathbb{E}_{(w,l)\sim\mu}\,\mathbb{E}_{\widetilde{w},\widetilde{l}\sim\mathsf{R}_\gamma}\Big[\big(1-\sigma(\beta(u_w-u_l))\big)\,\big(\mathbf{e}_{\widetilde{w}} - \mathbf{e}_{\widetilde{l}}\big)\Big], \tag{13}$$

where, given (w, l), the endpoints $\widetilde{w} \sim \mathsf{R}_\gamma(\cdot \mid w)$ and $\widetilde{l} \sim \mathsf{R}_\gamma(\cdot \mid l)$ are drawn independently.

Proof. By (12), it holds that $d(u) = -\gamma^{-1}\mathsf{R}_\gamma\nabla\mathcal{L}(u)$. Recalling (6) and $n_e = N\mu(e)$ gives

$$d(u) = \frac{\beta N}{\gamma}\sum_{e=(w,l)\in\mathcal{E}} \mu(e)\,\big[1-\sigma(\beta(u_w-u_l))\big]\,\big(\mathsf{R}_\gamma\mathbf{e}_w - \mathsf{R}_\gamma\mathbf{e}_l\big).$$

The vector $\mathsf{R}_\gamma \mathbf{e}_y$ is the y-th column of R_γ, and its z-th entry $[\mathsf{R}_\gamma]_{zy}$ equals $[\mathsf{R}_\gamma]_{yz} = \mathsf{R}_\gamma(z \mid y)$, as R_γ is symmetric by (12); hence $\mathsf{R}_\gamma \mathbf{e}_y = \sum_{z \in \mathcal{Y}} \mathsf{R}_\gamma(z \mid y)\, \mathbf{e}_z = \mathbb{E}[\mathbf{e}_{\widetilde{y}}]$ with $\widetilde{y} \sim \mathsf{R}_\gamma(\cdot \mid y)$. Substituting $y = w$ and $y = l$ and reading the sum over $e \in \mathcal{E}$ as an expectation over $(w, l) \sim \mu$ gives (13). □

The above theorem reveals that sampling the comparison $(w, l) \sim \mu$ and the corresponding endpoint pair $(\widetilde{w}, \widetilde{l}) \sim \mathsf{R}_\gamma$ gives the following unbiased estimator

$$\widehat{d}(u) \;=\; \frac{\beta N}{\gamma}\left(1 - \sigma(\beta(u_w - u_l))\right)\left(\mathbf{e}_{\widetilde{w}} - \mathbf{e}_{\widetilde{l}}\right), \qquad \mathbb{E}\left[\widehat{d}(u)\right] = d(u). \tag{14}$$

5.2 Extension to online learning

The preceding discussion relies on the objective (5), where the comparisons and their weights are fixed in advance. A subsequent extension is to consider the on-policy sampling, where the responses are generated by the current policy π, producing the following objective

$$\mathcal{L}(u; \pi) = -\mathbb{E}_{y,z \sim \pi}\left[\log \sigma\big(\beta(u(y) - u(z))\big)\right]. \tag{15}$$

The variable u is now a function assigning a score $u(y)$ to every possible response y, and the functional $\mathcal{L}(u; \pi)$ maps the score function to a scalar loss. Letting g be a perturbation direction and considering the inner product $\langle g, h \rangle_\pi = \mathbb{E}_{y \sim \pi}[g(y)h(y)]$, the first and second derivatives along the perturbation are

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{L}(u + \varepsilon g; \pi)\Big|_{\varepsilon=0} = \langle \nabla \mathcal{L}(u; \pi), g \rangle_\pi, \qquad \frac{\mathrm{d}^2}{\mathrm{d}\varepsilon^2}\mathcal{L}(u + \varepsilon g; \pi)\Big|_{\varepsilon=0} = \langle g, \nabla^2 \mathcal{L}(u; \pi) g \rangle_\pi.$$

Consequently, we can characterize the operation of the Hessian on g as follows,

$$\big(\nabla^2 \mathcal{L}(u; \pi)\, g\big)(y) \;=\; 2\beta^2\, \mathbb{E}_{z \sim \pi}\Big[\sigma\big(\beta[u(y) - u(z)]\big)\big[1 - \sigma\big(\beta[u(y) - u(z)]\big)\big]\,\big(g(y) - g(z)\big)\Big], \tag{16}$$

for any y in the response space.

The following proposition represents the regularized inverse of the online Hessian through the endpoint of a random walk. Section 5.3 uses this walk as the online oracle of PDPO.

Proposition 5.2. *Given a response y, set $X_0 = y$. At step $k = 0, 1, 2, \ldots$, stop and output X_k with probability $4\gamma/(4\gamma + \beta^2)$; otherwise draw $z \sim \pi$ and set $X_{k+1} = z$ with probability $4\,\sigma\big(\beta[u(X_k) - u(z)]\big)\big[1 - \sigma\big(\beta[u(X_k) - u(z)]\big)\big] \in (0, 1]$, and $X_{k+1} = X_k$ otherwise. Let $\widetilde{y}$ denote the response output when the walk stops. Then, we have*

$$\mathbb{E}\big[g(\widetilde{y})\big] \;=\; \big[2\gamma\big(2\gamma + \nabla^2 \mathcal{L}(u; \pi)\big)^{-1} g\big](y).$$

Proof. See Section B. □

5.3 The PDPO algorithm

With the preconditioned direction developed in Sections 5.1 and 5.2, we now adapt it into an update of the policy parameters θ, which determine the scores through $u_\theta(y) = \log \pi_\theta(y \mid x) - \log \pi_{\mathrm{ref}}(y \mid x)$. Therefore, it is natural to take into account J_θ, the Jacobian of the map $\theta \mapsto (u_\theta(y))_{y \in \mathcal{Y}}$. Consequently, we propose a preconditioned method for DPO (PDPO): each step obtains a preconditioned direction d, and updates $\theta \leftarrow \theta + \eta J_\theta^\top d$; see Algorithm 1. To compute d, we provide three oracles, chosen by the training scenario. In the offline scenario, the oracle either solves the linear system in (10) explicitly or samples the estimator (14) through the walk of Theorem 5.1; in the online scenario (15), the oracle samples through the walk of Proposition 5.2. These oracles are presented in Algorithms 2 to 4, respectively, and more details are referred to Section C.

6 Acceleration of DPO: Local Quadratic Convergence

Taking $u = 0$ in the Hessian formula (7) gives

$$\nabla^2 \mathcal{L}(0) = \frac{\beta^2}{4} \sum_{e=(w,l) \in \mathcal{E}} n_e (\mathbf{e}_w - \mathbf{e}_l)(\mathbf{e}_w - \mathbf{e}_l)^\top,$$

which is the Laplacian of the comparison graph with edge weights n_e, scaled by $\beta^2/4$. Under Assumption 4.1, the eigenvalues of $\nabla^2\mathcal{L}(0)$ satisfy $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_K$ by Proposition 4.2. We therefore define the condition number of problem (5) by $\kappa_{\mathcal{G}} := \lambda_K/\lambda_2$, which, in general, hinders the convergence rate of first-order gradient methods (Nesterov, 2018).

The following theorem shows that the exact Newton step of Proposition 4.3, the $\gamma \to 0$ limit of (10), converges quadratically to a stationary point $u^\star$ of $\mathcal{L}$, that is, $\nabla\mathcal{L}(u^\star) = 0$, independent of the condition number $\kappa_{\mathcal{G}}$. We measure the error using $\|v\|_{\nabla^2\mathcal{L}(u^\star)} = (v^\top\nabla^2\mathcal{L}(u^\star)v)^{1/2}$, which ignores a common shift on response scores by Proposition 4.2. For each comparison $e = (w, l)$, let $\Delta_e := \beta\big[(u_w - u_l) - (u_w^\star - u_l^\star)\big]$ denote the error of the reward difference.

Theorem 6.1 (Local quadratic convergence)**.** *Let Assumption 4.1 hold and let $u^\star \in \mathbb{R}^K$ satisfy $\nabla\mathcal{L}(u^\star) = 0$. There exists an open neighborhood $\mathcal{U} \subseteq \mathbb{R}^K$ of $u^\star$ such that, for every $u \in \mathcal{U}$, the exact Newton iterate $u^+ = u - (\nabla^2\mathcal{L}(u))^\dagger\nabla\mathcal{L}(u)$ of Proposition 4.3 satisfies*

$$\big\|u^+ - u^\star\big\|_{\nabla^2\mathcal{L}(u^\star)} \;\leq\; c_\star\left(\max_{e\in\mathcal{E}}|\Delta_e|\right)\|u - u^\star\|_{\nabla^2\mathcal{L}(u^\star)} \;+\; O\big(\|u - u^\star\|^3\big), \tag{17}$$

with constant $c_\star = \max_{(w,l)\in\mathcal{E}}\big|\sigma\big(\beta(u_w^\star - u_l^\star)\big) - \frac{1}{2}\big| \leq \frac{1}{2}$, which does not involve $\kappa_{\mathcal{G}}$.

Proof. See Section D.2. □

Since $\max_{e\in\mathcal{E}}|\Delta_e| = O(\|u - u^\star\|)$, the bound (17) reveals the locally quadratic convergence rate of the update along the Newton direction.

The estimator (14) has the mean $d(u)$ of (10) rather than the Newton step $\bar{\xi} = -\big(\nabla^2\mathcal{L}(u)\big)^\dagger\nabla\mathcal{L}(u)$ of (9). With a slight abuse of notation, let $\widehat{d}(u)$ denote the average of M independent samples of (14), let $\zeta := \|\widehat{d}(u) - \bar{\xi}\|_{\nabla^2\mathcal{L}(u)}/\|\bar{\xi}\|_{\nabla^2\mathcal{L}(u)}$ be its relative error, with ζ^t its value at the t-th iterate u^t, and let $\lambda_2(\nabla^2\mathcal{L}(u))$ be the smallest nonzero eigenvalue of $\nabla^2\mathcal{L}(u)$. The following proposition, based on the theory of inexact Newton methods (Dembo et al., 1982), bounds ζ and derives the convergence rates.

Proposition 6.2. *Let Assumption 4.1 hold and let $u^\star \in \mathbb{R}^K$ satisfy $\nabla\mathcal{L}(u^\star) = 0$. Then*

$$\zeta \leq \frac{\gamma}{\gamma + \lambda_2(\nabla^2\mathcal{L}(u))} + \frac{\big\|\widehat{d}(u) - d(u)\big\|_{\nabla^2\mathcal{L}(u)}}{\|\bar{\xi}\|_{\nabla^2\mathcal{L}(u)}}, \qquad \mathbb{E}\,\big\|\widehat{d}(u) - d(u)\big\|^2_{\nabla^2\mathcal{L}(u)} = O(1/M). \tag{18}$$

Moreover, given $c \in (0, 1)$, there exists a neighborhood $\mathcal{U}$ of $u^\star$ such that, if $u^0 \in \mathcal{U}$ and $\zeta^t \leq c$ for all t, the inexact Newton iterates $u^{t+1} = u^t + \widehat{d}(u^t)$ converge to $u^\star$ linearly. If, in addition, γ is annealed to zero and M is increased such that $\zeta^t \to 0$, the convergence is superlinear, and if $\zeta^t = O(\|\nabla\mathcal{L}(u^t)\|)$, the convergence is quadratic.

Proof. See Section D.3. □

7 Experiments

We evaluate the proposed method on synthetic problems, where the known minimizer allows us to measure the convergence rate, and on the fine-tuning task, where we measure preference-ranking accuracy and the win rate of generated responses. Details are referred to Section F.

7.1 Synthetic problems

Setup. To evaluate the convergence exactly, we begin by building DPO problems with a known minimizer, as in the simulations of Shi et al. (2025). There are K responses $\mathcal{Y} = \{1, 2, \ldots, K\}$ with known rewards $r_y = (y - 1)/(K - 1)$ for all $y \in \mathcal{Y}$, and a comparison graph determines which pairs are compared. For each compared edge $\{i, j\}$, we include both (i, j) and (j, i) in $\mathcal{E}$ and set the weights in objective (5) to $n_{(i,j)} = \sigma(r_i - r_j)$ and $n_{(j,i)} = \sigma(r_j - r_i)$, which models a probabilistic preference: response i is preferred to response j with probability $\sigma(r_i - r_j)$, and vice

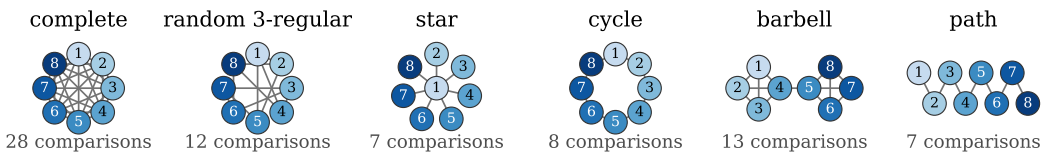


Figure 1: The six comparison graphs, drawn with $K = 8$. Each edge represents a comparison.

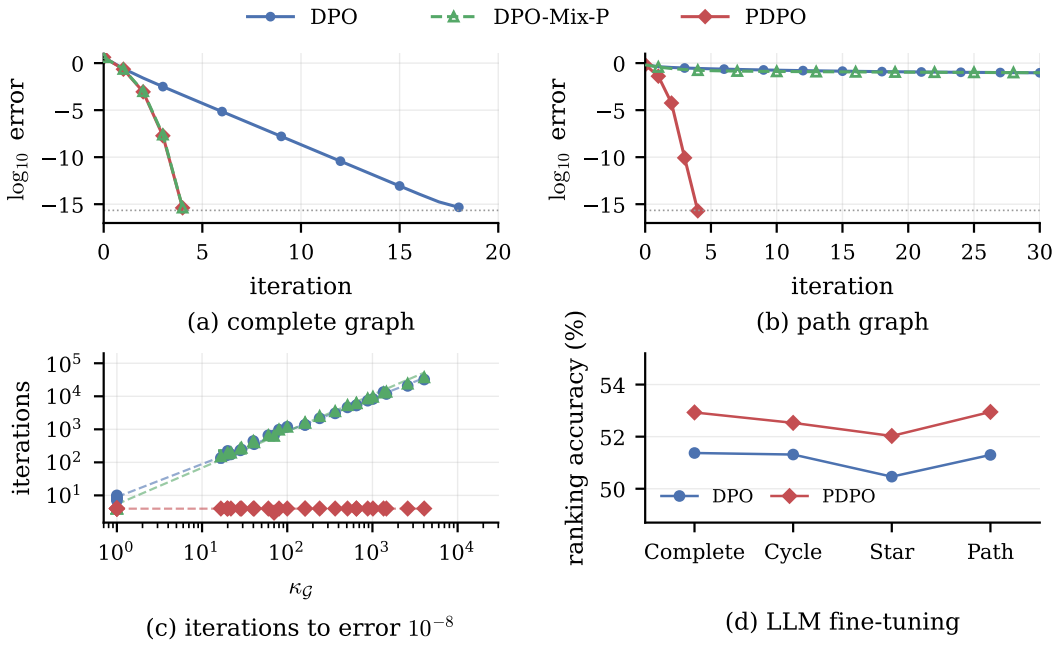


Figure 2: (a, b) Error against iteration with $K = 20$, under the complete and path graphs, respectively. (c) Iterations to reach 10^{-8} for six graph types at $K \in \{20, 40, 60, 80, 100\}$, averaged over five seeds. (d) Ranking accuracy of DPO and PDPO on the 4096 evaluation pairs after fine-tuning Qwen2.5-1.5B-Instruct in the offline setting (Section 7.2).

versa. Each pair's contribution is minimized when the model matches its preference probability, that is, $\sigma(\beta(u_i - u_j)) = \sigma(r_i - r_j)$, or equivalently $\beta(u_i - u_j) = r_i - r_j$. On a connected graph, the minimizers are therefore exactly $u^\star = r/\beta + c\mathbf{1}$, for all $c \in \mathbb{R}$.

The graphs, drawn in Figure 1, illustrate different geometry of the comparison. The complete graph compares every pair. The random 3-regular graph compares each response with exactly three others, chosen at random, and hence has $3K/2$ comparisons. The star graph compares response 1 with every other response and no other pair. The cycle graph compares each response y with $y + 1$, and response K with response 1. The barbell graph splits the responses by reward into a lower half $1, \ldots, K/2$ and an upper half $K/2+1, \ldots, K$, compares every pair within each group, and adds the single comparison of $K/2$ and $K/2 + 1$. The path compares only responses with adjacent indices.

The total weight of each compared pair is $n_{(i,j)} + n_{(j,i)} = 1$. Evaluating the Hessian (7) at $u = 0$ gives the condition number $\kappa_{\mathcal{G}} = \lambda_K/\lambda_2$. Taking $K = 20$ for instance, the condition numbers are 1, 16.6, 20, 40.9, 70 and 161 for the mentioned six graphs, respectively. We test on all six graph types at $K \in \{20, 40, 60, 80, 100\}$ with $\beta = 3$. For each configuration, all methods share the initialization $u^0 = v - K^{-1}(\mathbf{1}^\top v)\mathbf{1}$, where $v \sim \mathcal{N}(0, 0.1^2 I)$.

Methods and metric. We compare PDPO, which takes the Newton direction (9) by solving the linear system as in Algorithm 2 with $\gamma = 0$, with DPO, gradient descent on objective (5), and

Table 1: Ranking accuracy on 4096 pairs and generation win rate on 512 AlpacaEval prompts, averaged over five seeds.

Setting	Ranking accuracy (%)			Generation
	DPO	PDPO	Improvement	Win rate (%)
Offline, path	51.30	52.95	+1.65	55.37
Offline, complete	51.37	52.93	+1.56	51.07
Offline, cycle	51.31	52.53	+1.22	51.27
Offline, star	50.46	52.03	+1.56	52.15
Online	50.08	54.46	+4.38	52.74

with DPO-Mix-P of Shi et al. (2025). Including its sampling coefficients gives the effective weight $1/\omega(m_e) = 2(1 + \cosh(m_e))$ on each compared pair. DPO-Mix-P and DPO use a tuned step on every graph; see Section F.1. The error is reported by $\|u - u^\star\|_{\nabla^2 \mathcal{L}(u^\star)}$.

Results and analysis. On the complete graph, DPO-Mix-P and PDPO both reach error below 10^{-15} in four iterations (Figure 2(a)), which verifies the quadratic convergence rates. On the other graphs, PDPO retains rapid convergence, whereas DPO and DPO-Mix-P slow down as the graph becomes more ill-conditioned (Figure 2(b) and Figure 4). Across graph types and sizes, their iteration counts increase with $\kappa_{\mathcal{G}}$, while those of PDPO remain nearly constant (Figure 2(c)). In general, the rate of gradient descent depends on the condition number (Nesterov, 2018); DPO-Mix-P removes this dependence only on the complete graph (Section E), whereas PDPO converges locally quadratically under every connected graph (Theorem 6.1).

7.2 Language model

Setup. We fine-tune Qwen2.5-1.5B-Instruct with $\beta = 0.1$ on 256 prompts of UltraFeedback (Cui et al., 2024); the initial model also serves as the frozen reference π_{ref}. Each prompt comes with two to five distinct responses, sampled from the initial policy and scored by a reward model. Each method takes 64 updates on minibatches of four prompts. We report two metrics. Ranking accuracy on 4096 response pairs measures agreement between the order of the scores u_θ and that of the evaluation reward model. For generation, models trained via the respective methods answer the same 512 prompts sampled from AlpacaEval (Li et al., 2023), and GPT-5.5 judges each pair, and we report the win rate of PDPO over DPO; Section F.2 gives the details.

Offline setting. We connect the responses of each prompt by a path, a complete graph, a cycle or a star, and label every comparison by the sign of the reward difference, with ties labeled $\frac{1}{2}$. We compare PDPO invoking the preconditioned direction of Algorithm 2, with the baseline DPO, which takes the gradient of the loss averaged over all the comparison pairs. Both use AdamW, with learning rate 8×10^{-6} for DPO and 2×10^{-5} for PDPO, selected according to a validation set.

Online setting. When each prompt has only one labeled pair, DPO updates on this pair alone. PDPO estimates the preconditioned direction by random walks following Algorithm 4.

The overall performance is summarized in Table 1, which reveals that PDPO gains higher ranking accuracy and win rate, facilitating the training of the direct preference optimization problem.

8 Conclusion and Perspective

This work shows that the regularized Newton direction of DPO is an expectation of first-order information over random walks on the comparison graph (Theorem 5.1 and Proposition 5.2), and builds PDPO on this characterization. On a connected comparison graph, the proposed method enjoys favorable convergence property. As Yang et al. (2025) did for RLHF, this work characterizes second-order information exactly via first-order samples. Whether this technique extends to reinforcement learning problems with comparison structures in general, most directly to the variants of DPO such as (Azar et al., 2024; Hong et al., 2024; Meng et al., 2024; Ethayarajh et al., 2024), is a promising future work.

AI USE STATEMENT

In this work, we used generative AI tools for the following tasks: assisting in writing and checking proofs, giving feedback on the design of the experiments, assisting in implementing the methods and the experiment code. Additionally, we used generative AI tools for drafting and editing parts of the paper for readability, searching and summarizing the literature. We have reviewed all AI-assisted work: the authors checked the proofs, verified each citation against its source, and checked the code of the experiments. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The assumptions and complete proofs of the theoretical results are given in Sections A, B and D, and Section C presents the details of the proposed algorithm. Section F gives the details of the experiments.

REFERENCES

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Rémi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2024.

Francis Bach. Self-concordant analysis for logistic regression. *Electronic Journal of Statistics*, 4: 384–414, 2010.

Michele Benzi. Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics*, 182(2):418–477, 2002. doi: 10.1006/jcph.2002.7176.

Nicolas Boumal. *An Introduction to Optimization on Smooth Manifolds*. Cambridge University Press, 2023. doi: 10.1017/9781009166164.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *Transactions on Machine Learning Research*, 2023.

Gayathri Neela Chandran, Sai Soumya Nalli, Sruthi Gorantla, Amit Deshpande, and Anand Louis. The role of preference data and unembeddings in the convergence rate of DPO. In *2nd Workshop on Aligning Reinforcement Learning Experimentalists and Theorists (ARLET 2025)*, 2025. https://openreview.net/forum?id=QdeVcEX8y2.

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*, 2017.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. UltraFeedback: Boosting language models with scaled AI feedback. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 9722–9744. PMLR, 2024. URL https://proceedings.mlr.press/v235/cui24f.html.

Ron S. Dembo, Stanley C. Eisenstat, and Trond Steihaug. Inexact Newton methods. *SIAM Journal on Numerical Analysis*, 19(2):400–408, 1982.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Model alignment as prospect theoretic optimization. In *International Conference on Machine Learning (ICML)*, 2024.

Bin Gao, Renfeng Peng, and Ya-xiang Yuan. Optimization on product manifolds under a preconditioned metric. *SIAM Journal on Matrix Analysis and Applications*, 46(3):1816–1845, 2025. doi: 10.1137/24M1643773.

Jiafan He, Huizhuo Yuan, and Quanquan Gu. Accelerated preference optimization for large language model alignment. In *NeurIPS 2024 Workshop on Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*, 2024.

Jiwoo Hong, Noah Lee, and James Thorne. ORPO: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 11170–11189, 2024.

Juno Kim, Jihun Yun, Jason D. Lee, and Kwang-Sung Jun. Coverage improvement and fast convergence of on-policy preference learning. In *International Conference on Machine Learning (ICML)*, 2026.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. AlpacaEval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 2023.

Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989. doi: 10.1007/BF01589116.

Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, Peter J. Liu, and Xuanhui Wang. LiPO: Listwise preference optimization through learning-to-rank. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2404–2420, 2025.

Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, and Yang Liu. Skywork-Reward-V2: Scaling preference data curation via human-AI synergy. In *International Conference on Learning Representations (ICLR)*, pp. 133805–133838, 2026.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *International Conference on Machine Learning (ICML)*, 2015.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *Advances in Neural Information Processing Systems*, volume 37, pp. 124198–124235, 2024.

Bamdev Mishra and Rodolphe Sepulchre. Riemannian preconditioning. *SIAM Journal on Optimization*, 26(1):635–660, 2016. doi: 10.1137/140970860.

Yurii Nesterov. *Lectures on Convex Optimization*. Springer, 2nd edition, 2018.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, New York, 2 edition, 2006. doi: 10.1007/978-0-387-40065-5.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

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*, volume 36, pp. 53728–53741, 2023.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, PA, 2 edition, 2003. doi: 10.1137/1.9780898718003.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Ruizhe Shi, Runlong Zhou, and Simon S. Du. The crucial role of samplers in online direct preference optimization. In *International Conference on Learning Representations (ICLR)*, pp. 39255–39285, 2025.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18990–18998, 2024.

Tianxiao Sun and Quoc Tran-Dinh. Generalized self-concordant functions: a recipe for Newton-type methods. *Mathematical Programming*, 178(1–2):145–213, 2019.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10582–10592, 2024.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is DPO superior to PPO for LLM alignment? a comprehensive study. In *International Conference on Machine Learning (ICML)*, 2024.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Yan Yang, Bin Gao, and Ya-xiang Yuan. Bilevel reinforcement learning via the development of hyper-gradient without lower-level convexity. In Yingzhen Li, Stephan Mandt, Shipra Agrawal, and Emtiyaz Khan (eds.), *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 258 of *Proceedings of Machine Learning Research*, pp. 4780–4788. PMLR, 2025. URL https://proceedings.mlr.press/v258/yang25g.html.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. RRHF: Rank responses to align language models with human feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10935–10950, 2023.

A Proofs for the geometry of the comparison graph

This section presents proofs for Propositions 4.2 and 4.3 of Section 4. Throughout the appendix, for a comparison $e = (w, l)$, we let $b_e = \mathbf{e}_w - \mathbf{e}_l$ and write $m_e(u) = \beta(u_w - u_l) = \beta b_e^\top u$ for its reward difference, and we write $\omega(t) = \sigma(t)[1 - \sigma(t)]$, such that $\beta^2 n_e\, \omega(m_e(u))$ is the weight of the comparison e in the Hessian formula (7).

A.1 Kernel of the Hessian

The proof of Proposition 4.2 writes the quadratic form of the Hessian as a weighted sum of squared differences across the comparisons, which vanishes only at vectors that take the same value at the two ends of every comparison.

Proof of Proposition 4.2. For $v \in \mathbb{R}^K$, it follows from the Hessian formula (7) that

$$v^\top \nabla^2 \mathcal{L}(u) v = \beta^2 \sum_{e=(w,l)\in\mathcal{E}} n_e\, \omega(m_e(u))\, (v_w - v_l)^2 \geq 0, \tag{19}$$

and hence $\nabla^2 \mathcal{L}(u)$, which is symmetric as a weighted sum of the matrices $b_e b_e^\top$, is positive semidefinite. Taking $v = \mathbf{1}$ in the quadratic form (19) gives $\mathbf{1}^\top \nabla^2 \mathcal{L}(u) \mathbf{1} = 0$, which, together with positive semidefiniteness, implies $\nabla^2 \mathcal{L}(u)\mathbf{1} = 0$. Since $n_e > 0$ and $\omega > 0$, every comparison carries a positive weight $n_e \omega(m_e(u))$, and thus $v^\top \nabla^2 \mathcal{L}(u) v = 0$ forces $v_w = v_l$ for every $(w, l) \in \mathcal{E}$. Under

Assumption 4.1, any two responses are joined by a path of comparisons, and hence v is constant. Therefore, $\ker \nabla^2\mathcal{L}(u) = \operatorname{span}\{\mathbf{1}\}$. By symmetry, $\operatorname{range}(\nabla^2\mathcal{L}(u)) = (\ker \nabla^2\mathcal{L}(u))^\perp = \mathcal{H}$, and the restriction of $\nabla^2\mathcal{L}(u)$ to $\mathcal{H}$ is positive definite, since every $v \in \mathcal{H}$ with $v^\top \nabla^2\mathcal{L}(u) v = 0$ lies in $\operatorname{span}\{\mathbf{1}\} \cap \mathcal{H} = \{0\}$. □

A.2 Newton step on the quotient space

The quotient Newton equation in Proposition 4.3 involves the gradient and the Hessian of $\overline{\mathcal{L}}$ on the quotient space $\mathcal{M}$, of which we compute the representations in $\mathbb{R}^K$ following Boumal (2023, Chapter 9). The additive group $\mathbb{R}$ acts on $\mathbb{R}^K$ by the translations $u \mapsto u + c\mathbf{1}$, and the orbits of this action are the classes $[u] = \{v \in \mathbb{R}^K : u + c\mathbf{1}, c \in \mathbb{R}\}$. This action is smooth and free, and it is proper by Boumal (2023, Proposition 9.60); hence, $\mathcal{M}$ is a smooth quotient space of dimension $K - 1$ with the quotient map $\pi : u \mapsto [u]$ (Boumal, 2023, Theorem 9.18). At $u \in \mathbb{R}^K$, the vertical space $\ker \mathrm{D}\pi(u)$ is the tangent space $\operatorname{span}\{\mathbf{1}\}$ of the class $[u]$, and the horizontal space is the orthogonal complement, the space $\mathcal{H}$ defined in Section 4 (Boumal, 2023, Definition 9.24). The restriction of $\mathrm{D}\pi(u)$ to $\mathcal{H}$ is a bijection onto the tangent space $\mathrm{T}_{[u]}\mathcal{M}$, and its inverse maps $\xi \in \mathrm{T}_{[u]}\mathcal{M}$ to the horizontal lift $\overline{\xi} = \operatorname{lift}_u(\xi)$ of Section 4. Since each translation $u \mapsto u + c\mathbf{1}$ has the identity as its differential, the inner product $\langle \xi, \zeta \rangle_{[u]} = \langle \overline{\xi}, \overline{\zeta} \rangle$ on $\mathrm{T}_{[u]}\mathcal{M}$, inherited from $\mathbb{R}^K$ through the horizontal lifts, does not depend on the representative u (Boumal, 2023, Definition 9.37 and Theorem 9.38). For a smooth function f on $\mathcal{M}$, the gradient and the Hessian of f with respect to this inner product then lift as

$$\operatorname{lift}_u\big(\operatorname{grad} f([u])\big) = \nabla(f \circ \pi)(u), \qquad \operatorname{lift}_u\big(\operatorname{Hess} f([u])[\xi]\big) = \Pi_{\mathcal{H}} \nabla^2(f \circ \pi)(u)\, \overline{\xi} \tag{20}$$

for every $\xi \in \mathrm{T}_{[u]}\mathcal{M}$, where $\Pi_{\mathcal{H}}$ denotes the orthogonal projection onto $\mathcal{H}$ (Boumal, 2023, Propositions 9.39 and 9.45).

The proof of Proposition 4.3 uses the formulas (20) to lift the quotient Newton equation to a linear system on $\mathcal{H}$, on which $\nabla^2\mathcal{L}(u)$ is invertible by Proposition 4.2, and relates the solution to a step containing the pseudoinverse of $\nabla^2\mathcal{L}(u)$ in the original space $\mathbb{R}^K$.

Proof of Proposition 4.3. Since $\mathcal{L}$ is constant on each class $[u]$, the function $\overline{\mathcal{L}}$ is well defined, $\overline{\mathcal{L}} \circ \pi = \mathcal{L}$, and $\overline{\mathcal{L}}$ is smooth because $\mathcal{L}$ is (Boumal, 2023, Theorem 9.21). By the formulas (20) with $f = \overline{\mathcal{L}}$, the horizontal lift of $\operatorname{grad} \overline{\mathcal{L}}([u])$ is $\nabla\mathcal{L}(u)$, which lies in $\mathcal{H}$. The horizontal lift of $\operatorname{Hess} \overline{\mathcal{L}}([u])[\xi]$ is $\Pi_{\mathcal{H}} \nabla^2\mathcal{L}(u)\overline{\xi}$, which equals $\nabla^2\mathcal{L}(u)\overline{\xi}$, since $\nabla^2\mathcal{L}(u)$ maps $\mathcal{H}$ into $\mathcal{H}$ by Proposition 4.2. Hence, the quotient Newton equation lifts to the linear system

$$\nabla^2\mathcal{L}(u)\overline{\xi} = -\nabla\mathcal{L}(u), \qquad \overline{\xi} \in \mathcal{H},$$

which has a unique solution, as the restriction of $\nabla^2\mathcal{L}(u)$ to $\mathcal{H}$ is positive definite by Proposition 4.2. Noting that $\nabla\mathcal{L}(u) \in \mathcal{H}$, the vector $-\big(\nabla^2\mathcal{L}(u)\big)^\dagger \nabla\mathcal{L}(u)$ lies in $\mathcal{H}$ and solves the linear system, which gives the lift (9).

It remains to represent the quotient Newton update in $\mathbb{R}^K$. Since $\pi(u + c\mathbf{1} + v) = \pi(u + v)$, it holds that $\mathrm{D}\pi(u + c\mathbf{1}) = \mathrm{D}\pi(u)$, and hence we have $\operatorname{lift}_{u+c\mathbf{1}}(\xi) = \operatorname{lift}_u(\xi)$ for all $c \in \mathbb{R}$. The step $s \mapsto u + s$ in $\mathbb{R}^K$ therefore induces a well-defined step $[u] \mapsto [u + \operatorname{lift}_u(\xi)]$ on $\mathcal{M}$, with which the Newton update reads $[u] \mapsto [u + \overline{\xi}]$ (Boumal, 2023, Section 9.12). Taking $u^+ = u + \overline{\xi}$ at any representative u in $[u]$ gives the iteration in Proposition 4.3. □

B Random-walk representation of the online Hessian

This section proves Proposition 5.2, on which the online oracle of Section C rests. Conditioning on the first step y of the walk, the proof of Proposition 5.2 shows that the function $h(y) = \mathbb{E}[g(\widetilde{y}) \mid X_0 = y]$ solves the linear system $(2\gamma + \nabla^2\mathcal{L}(u; \pi))h = 2\gamma g$.

Proof of Proposition 5.2. Since $\omega(t) = \sigma(t)[1 - \sigma(t)] \leq \frac{1}{4}$, the acceptance probabilities $4\omega(\beta[u(y) - u(z)])$ lying in $(0, 1]$ are well-defined. Each step stops with probability $4\gamma/(4\gamma + \beta^2) > 0$, and hence the walk stops after finitely many steps with probability one. Given a bounded function g on the responses, we define $h(y) = \mathbb{E}[g(\widetilde{y}) \mid X_0 = y]$, the expectation taken over the random walk

started at $X_0 = y$. At step 0, the walk stops and outputs y with probability $4\gamma/(4\gamma+\beta^2)$. Otherwise, it samples $z \sim \pi$ and sets $X_1 = z$ with probability $4\omega(\beta[u(y)-u(z)])$ and $X_1 = y$ otherwise. Since the rule of each step depends only on the current response, we have $\mathbb{E}[g(\widetilde{y}) \mid X_0 = y,\, X_1] = h(X_1)$. Therefore,

$$h(y) = \tfrac{4\gamma}{4\gamma+\beta^2}\, g(y) + \tfrac{\beta^2}{4\gamma+\beta^2}\, \mathbb{E}_{z\sim\pi}\big[4\omega(\beta[u(y)-u(z)])\, h(z) + \big(1 - 4\omega(\beta[u(y)-u(z)])\big)\, h(y)\big].$$

By the Hessian formula (16), the expectation on the right-hand side equals $h(y) - \frac{2}{\beta^2}(\nabla^2\mathcal{L}(u;\pi)h)(y)$. Multiplying both sides by $4\gamma+\beta^2$ gives $4\gamma h + 2\nabla^2\mathcal{L}(u;\pi)h = 4\gamma g$. Note that $\langle g, \nabla^2\mathcal{L}(u;\pi)g\rangle_\pi = \beta^2\mathbb{E}_{y,z\sim\pi}[\omega(\beta[u(y)-u(z)])(g(y)-g(z))^2] \geq 0$, and thus $\nabla^2\mathcal{L}(u;\pi)$ is positive semidefinite and $2\gamma+\nabla^2\mathcal{L}(u;\pi)$ is invertible. Therefore, $h = 2\gamma(2\gamma+\nabla^2\mathcal{L}(u;\pi))^{-1}g$. □

C THE PDPO ALGORITHM

This section serves as a supplement to Section 5.3. We first present how the preconditioned direction with respect to the score variable u induces an update of the policy parameter θ, and then produce PDPO as one main loop that invokes one of three oracles for the preconditioned direction.

C.1 FROM SCORE DIRECTIONS TO PARAMETER UPDATES

The loss depends on the parameters θ through the scores $u_\theta(y) = \log\pi_\theta(y \mid x) - \log\pi_{\mathrm{ref}}(y \mid x)$. Let J_θ denote the Jacobian of the map $\theta \mapsto (u_\theta(y))_y$. According to the chain rule, PDPO applies the adjoint of J_θ to a preconditioned direction to update θ.

This subsection discusses firstly on a finite comparison graph and then the online objective. The row of J_θ for a response y is $\nabla_\theta u_\theta(y)^\top$, and hence, for a direction d that is nonzero at finitely many responses to x,

$$J_\theta^\top d = \sum_y d(y)\, \nabla_\theta u_\theta(y).$$

For a response y, we write $\mathbb{1}_y$ for its indicator function, that is, $\mathbb{1}_y(z) = 1$ if $z = y$ and $\mathbb{1}_y(z) = 0$ otherwise. On a finite set of responses $y_1, \ldots, y_K$, where a direction d is identified with the vector $(d(y_k))_{k=1}^K \in \mathbb{R}^K$, the indicator $\mathbb{1}_y$ is the standard basis vector $\mathbf{e}_y$ used in the estimator (14).

Finite comparison graph. The chain rule gives $\nabla_\theta\mathcal{L}(u_\theta) = J_\theta^\top\nabla\mathcal{L}(u_\theta)$, and hence a DPO step moves θ along $-J_\theta^\top\nabla\mathcal{L}(u_\theta)$. PDPO moves θ along $J_\theta^\top d(u_\theta)$ instead, with d the regularized Newton direction (10). When $d(u_\theta)$ is replaced by the estimate (14), which has the expectation $d(u_\theta)$ by Theorem 5.1, the linearity of $J_\theta^\top$ shows that the update satisfies

$$\mathbb{E}[\theta^+ - \theta \mid \theta] = \eta J_\theta^\top d(u_\theta),$$

and averaging M independent estimates reduces the variance (Proposition 6.2).

Online objective. Write $\pi = \pi_\theta(\cdot \mid x)$ and $u = u_\theta$, and let $L^2(\pi)$ be the space of functions h on the responses with $\mathbb{E}_{y\sim\pi}[h(y)^2] < \infty$, with the inner product $\langle g, h\rangle_\pi = \mathbb{E}_{y\sim\pi}[g(y)h(y)]$. With two responses sampled from π and ordered as (w, l) by an annotator, the online loss is

$$\mathcal{L}_{\mathrm{lab}}(u;\pi) = -\mathbb{E}_{(w,l)}\big[\log\sigma\big(\beta[u(w)-u(l)]\big)\big]. \tag{21}$$

Its gradient g_π and Hessian H_π in $L^2(\pi)$ satisfy $H_\pi = \nabla^2\mathcal{L}(u;\pi)$ of (16). The walk of Proposition 5.2 thus realizes the self-adjoint operator R_π below, and the corresponding regularized Newton direction is d_π:

$$\mathsf{R}_\pi = 2\gamma(2\gamma I + H_\pi)^{-1}, \qquad d_\pi = -(2\gamma I + H_\pi)^{-1} g_\pi.$$

In $L^2(\pi)$, the transpose $J_\theta^\top$ is replaced by the adjoint J_π^* of J_θ, which satisfies

$$a^\top J_\pi^* h = \langle J_\theta a, h\rangle_\pi, \qquad J_\pi^* h = \mathbb{E}_{y\sim\pi}\big[h(y)\nabla_\theta u_\theta(y)\big].$$

With $\rho(w,l) = 1 - \sigma(\beta[u(w)-u(l)])$, differentiating the online loss (21) along a score perturbation h gives

$$\langle g_\pi, h\rangle_\pi = -\beta\,\mathbb{E}_{(w,l)}\big[\rho(w,l)(h(w)-h(l))\big]. \tag{22}$$

As H_π cannot be formed, the preconditioned direction is sampled: a walk of Proposition 5.2 is run from w and another from l, with endpoints $\widetilde{w}$ and $\widetilde{l}$, and θ moves along

$$\widehat{v} = J_\theta^\top \Big[\frac{\beta}{\gamma}\rho(w,l)\big(\mathbb{1}_{\widetilde{w}} - \mathbb{1}_{\widetilde{l}}\big)\Big] = \frac{\beta}{\gamma}\rho(w,l)\big[\nabla_\theta u_\theta(\widetilde{w}) - \nabla_\theta u_\theta(\widetilde{l})\big].$$

Applying Proposition 5.2 to the function $J_\theta a$ gives $\mathbb{E}[(J_\theta a)(\widetilde{y}) \mid X_0 = y] = (\mathsf{R}_\pi J_\theta a)(y)$. Conditioning on (w,l) and using the identity (22) and the self-adjointness of R_π, we obtain

$$\begin{aligned} a^\top \mathbb{E}[\widehat{v}] &= \frac{\beta}{\gamma}\mathbb{E}_{(w,l)}\big[\rho(w,l)\big((\mathsf{R}_\pi J_\theta a)(w) - (\mathsf{R}_\pi J_\theta a)(l)\big)\big] \\ &= -\frac{1}{\gamma}\langle g_\pi, \mathsf{R}_\pi J_\theta a\rangle_\pi = -\frac{1}{\gamma}\langle \mathsf{R}_\pi g_\pi, J_\theta a\rangle_\pi \\ &= 2\langle d_\pi, J_\theta a\rangle_\pi = 2a^\top J_\pi^* d_\pi. \end{aligned}$$

Since a is arbitrary, the update $\theta^+ = \theta + \eta\widehat{v}$ satisfies

$$\mathbb{E}[\theta^+ - \theta \mid \theta] = 2\eta J_\pi^* d_\pi. \tag{23}$$

C.2 Main loop and oracles

Algorithm 1 implements the update of Section C.1. It samples a prompt x, obtains a preconditioned direction d from an oracle, and moves θ along $J_\theta^\top d$; with a minibatch of prompts, the directions of the prompts are averaged.

The branch in Algorithm 1 results from the different scenarios. When the responses to x carry a finite comparison graph with comparisons $\mathcal{E}$ and weights n_e, as in objective (5), whether the comparisons come from a preference dataset or from responses sampled from the current policy and labeled before the step, two oracles apply. Algorithm 2 forms the gradient (6) and the Hessian (7) and solves the linear system in (10), which costs $O(K^3)$ operations and returns $d(u_\theta)$ exactly. Algorithm 3 runs the random walk of Section 5.1, which reads only the Hessian entries (11) at the visited responses, and returns the estimate (14). For the online objective (15), where the responses range over the outputs of the current policy and $\nabla^2\mathcal{L}$ cannot be formed explicitly, Algorithm 4 runs the walk of Proposition 5.2.

The two walk oracles return directions in the same fashion. Each samples a pair (w,l) with $w \succ l$, from the comparisons $\mathcal{E}$ in Algorithm 3 and from the current policy with a label of the annotator in Algorithm 4, runs one walk from w and one from l, with endpoints $\widetilde{w}$ and $\widetilde{l}$, and returns

$$d = c\,\big(1 - \sigma(\beta[u_\theta(w) - u_\theta(l)])\big)\big(\mathbb{1}_{\widetilde{w}} - \mathbb{1}_{\widetilde{l}}\big),$$

with $c = \frac{\beta N}{\gamma}$ or $c = \frac{\beta}{\gamma}$, respectively. As a remark, by Section C.1, the expected update of Algorithm 1 is $\eta J_\theta^\top d(u_\theta)$ with Algorithm 2 or Algorithm 3, and $2\eta J_\pi^* d_\pi$ with Algorithm 4, where the factor 2 is absorbed into the step size.

Algorithm 1 PDPO: the preconditioned method for DPO

Require: initial parameters θ, reference policy π_{ref}, KL coefficient β, regularization $\gamma \geq 0$ ($\gamma > 0$ for the walks), step size η, number of iterations T; for each prompt, a comparison graph on its responses or an annotator

for $t = 0, 1, \ldots, T-1$ **do**
  sample a prompt x
  if the responses to x carry a finite comparison graph **then**
    if the linear system in (10) is affordable **then**
      $d \leftarrow \textsc{Solve}(\theta, x)$ (Algorithm 2)
    else
      $d \leftarrow \textsc{GraphWalk}(\theta, x)$ (Algorithm 3)
    end if
  else
    $d \leftarrow \textsc{OnlineWalk}(\theta, x)$ (Algorithm 4)
  end if
  update $\theta \leftarrow \theta + \eta J_\theta^\top d$
end for

Algorithm 2 SOLVE(θ, x): explicit solve on a finite comparison graph

Require: responses $y_1, \dots, y_K$ to x, comparisons $\mathcal{E}$ with weights n_e

compute the scores $u = (u_\theta(y_k))_{k=1}^K$
form the gradient (6) and the Hessian (7) at u
solve $(\gamma I + \nabla^2 \mathcal{L}(u))d = -\nabla \mathcal{L}(u)$, taking the Newton step (9) if $\gamma = 0$
return d, with $d(y_k) = d_k$

Algorithm 3 GRAPHWALK(θ, x): random walk on a finite comparison graph

Require: responses $y_1, \dots, y_K$ to x, comparisons $\mathcal{E}$ with weights n_e and total weight $N = \sum_{e \in \mathcal{E}} n_e$

compute the scores $u = (u_\theta(y_k))_{k=1}^K$
sample a comparison $(w, l) \in \mathcal{E}$ with probability $n_{(w,l)}/N$
for $y_0 \in \{w, l\}$ **do**
  $y \leftarrow y_0$
  repeat
    compute the Hessian entries $[\nabla^2 \mathcal{L}(u)]_{yz}$, $z \neq y$, by (11)
    stop with probability $\gamma / (\gamma + [\nabla^2 \mathcal{L}(u)]_{yy})$; otherwise move to $z \neq y$ with probability $-[\nabla^2 \mathcal{L}(u)]_{yz} / (\gamma + [\nabla^2 \mathcal{L}(u)]_{yy})$
  until stopped
  $\widetilde{y}_0 \leftarrow y$
end for
return $\frac{\beta N}{\gamma} \big(1 - \sigma(\beta[u_w - u_l])\big) \big(\mathbf{e}_{\widetilde{w}} - \mathbf{e}_{\widetilde{l}}\big)$

Algorithm 4 ONLINEWALK(θ, x): random walk on the responses of the current policy

Require: an annotator of preferences

sample two responses independently from $\pi_\theta(\cdot \mid x)$ and label the pair by the annotator to get (w, l), $w \succ l$
for $y_0 \in \{w, l\}$ **do**
  $y \leftarrow y_0$
  repeat
    stop with probability $4\gamma / (4\gamma + \beta^2)$
    if not stopped, sample $z \sim \pi_\theta(\cdot \mid x)$ and set $y \leftarrow z$ with probability $4\,\sigma(\beta[u_\theta(y) - u_\theta(z)]) \big[1 - \sigma(\beta[u_\theta(y) - u_\theta(z)])\big]$; otherwise keep y unchanged
  until stopped
  $\widetilde{y}_0 \leftarrow y$
end for
return $\frac{\beta}{\gamma} \big(1 - \sigma(\beta[u_\theta(w) - u_\theta(l)])\big) \big(\mathbb{1}_{\widetilde{w}} - \mathbb{1}_{\widetilde{l}}\big)$

D PROOFS FOR THE LOCAL CONVERGENCE

This section proves Theorem 6.1 and Proposition 6.2. We introduce the notation: for a comparison $e = (w, l)$, $b_e = \mathbf{e}_w - \mathbf{e}_l$ and $m_e(u) = \beta(u_w - u_l) = \beta b_e^\top u$ is the reward difference, and $\omega(t) = \sigma(t)[1 - \sigma(t)]$.

D.1 PRELIMINARIES

The loss $\ell(t) = -\log \sigma(t)$ of a single comparison, with which objective (5) reads $\mathcal{L}(u) = \sum_{e \in \mathcal{E}} n_e \, \ell(m_e(u))$, satisfies $\ell'' = \omega$ and $\ell''' = \omega' = \omega(1 - 2\sigma)$, and hence $|\ell'''| \leq \ell''$, which is the scalar form of generalized self-concordance (Bach, 2010; Sun & Tran-Dinh, 2019). The following lemma turns this inequality into a comparison of the Hessians at two points.

Lemma D.1. *For all $u, u' \in \mathbb{R}^K$, with $\tau := \beta \max_{(w,l) \in \mathcal{E}} |(u'_w - u'_l) - (u_w - u_l)|$, it holds that*

$$\mathrm{e}^{-\tau} \, \nabla^2 \mathcal{L}(u) \preceq \nabla^2 \mathcal{L}(u') \preceq \mathrm{e}^{\tau} \, \nabla^2 \mathcal{L}(u). \tag{24}$$

Proof. The map $m \mapsto \log \omega(m) = \log \sigma(m) + \log(1 - \sigma(m))$ has the derivative $1 - 2\sigma(m) \in [-1, 1]$, and hence $|\log \omega(m') - \log \omega(m)| \le |m' - m|$, that is, $\mathrm{e}^{-|m'-m|} \le \omega(m')/\omega(m) \le \mathrm{e}^{|m'-m|}$. Since $|m_e(u') - m_e(u)| \le \tau$ for every $e \in \mathcal{E}$, it holds that $\omega(m_e(u')) \le \mathrm{e}^{\tau}\omega(m_e(u))$, and substituting this bound into each term of the quadratic form (19) gives $v^\top \nabla^2 \mathcal{L}(u')v \le \mathrm{e}^{\tau} v^\top \nabla^2 \mathcal{L}(u)v$ for all $v \in \mathbb{R}^K$, i.e., $\nabla^2 \mathcal{L}(u') \preceq \mathrm{e}^{\tau} \nabla^2 \mathcal{L}(u)$. The lower bound follows in the same way from $\omega(m_e(u')) \ge \mathrm{e}^{-\tau}\omega(m_e(u))$. □

Since the reward differences $m_e(u) = \beta b_e^\top u$ are affine in u, the bound $|\ell'''| \le \ell''$ also lifts to the third derivative of $\mathcal{L}$, with a constant that depends only on β, as the following lemma shows.

Lemma D.2 (Generalized self-concordance of $\mathcal{L}$)**.** *For all $u, h \in \mathbb{R}^K$, we have*

$$\left|\nabla^3 \mathcal{L}(u)[h, h, h]\right| \le \sqrt{2}\,\beta \,\|h\|_2 \,\left(h^\top \nabla^2 \mathcal{L}(u)\, h\right). \tag{25}$$

Proof. Differentiating $\mathcal{L}$ gives $\nabla^3 \mathcal{L}(u)[h, h, h] = \sum_e n_e \,\ell'''(m_e(u))\,(\beta b_e^\top h)^3$, and the bound $|\ell'''| \le \ell'' = \omega$ on every comparison yields

$$\left|\nabla^3 \mathcal{L}(u)[h, h, h]\right| \le \sum_e n_e\, \omega(m_e(u))\,(\beta b_e^\top h)^2 \,\left|\beta b_e^\top h\right|.$$

By the Cauchy–Schwarz inequality, the last term satisfies $\left|\beta b_e^\top h\right| = \beta\,|h_w - h_l| \le \beta\,\|b_e\|_2\,\|h\|_2 = \sqrt{2}\,\beta\,\|h\|_2$. Substituting it gives

$$\left|\nabla^3 \mathcal{L}(u)[h, h, h]\right| \le \sqrt{2}\,\beta\,\|h\|_2 \sum_e n_e\, \omega(m_e(u))\,(\beta b_e^\top h)^2 = \sqrt{2}\,\beta\,\|h\|_2 \,\left(h^\top \nabla^2 \mathcal{L}(u) h\right).$$

□

D.2 Local quadratic convergence of the Newton step

The proof of Theorem 6.1 tracks, through one Newton step from u to u^+, the error of the reward difference of each comparison e, that is, $\Delta_e = m_e(u) - m_e(u^\star) = \beta\big[(u_w - u_l) - (u_w^\star - u_l^\star)\big]$.

Proof of Theorem 6.1. Write $\epsilon = u - u^\star$, $m_e = m_e(u)$ and $m_e^\star = m_e(u^\star)$, such that $\Delta_e = m_e - m_e^\star = \beta b_e^\top \epsilon$. Let $B \in \mathbb{R}^{|\mathcal{E}| \times K}$ be the matrix with rows $b_e^\top$, which stacks the errors of all comparisons as $(\Delta_e)_{e\in\mathcal{E}} = \beta B\epsilon$, and let $D^\star = \mathrm{diag}\big(n_e \omega(m_e^\star)\big)$. The Hessian formula (7) at $u^\star$ then reads $\nabla^2 \mathcal{L}(u^\star) = \beta^2 B^\top D^\star B$.

Let $\epsilon^+ = u^+ - u^\star$. The fundamental theorem of calculus and $\nabla \mathcal{L}(u^\star) = 0$ give $\nabla \mathcal{L}(u) = \int_0^1 \nabla^2 \mathcal{L}(u^\star + t\epsilon)\epsilon\, dt$. Moreover, for a symmetric matrix A, the product $A^\dagger A$ is the orthogonal projection onto $\mathrm{range}(A) = (\ker A)^\perp$. By Proposition 4.2, the kernel of $\nabla^2 \mathcal{L}(u)$ is $\mathrm{span}\{\mathbf{1}\}$, and hence $\big(\nabla^2 \mathcal{L}(u)\big)^\dagger \nabla^2 \mathcal{L}(u) = I - \mathbf{1}\mathbf{1}^\top/K$, the orthogonal projection onto $\mathbf{1}^\perp$, which, together with $B\mathbf{1} = 0$, yields $B\epsilon = B\big(\nabla^2 \mathcal{L}(u)\big)^\dagger \nabla^2 \mathcal{L}(u)\epsilon$. Substituting both identities into $B\epsilon^+ = B\epsilon - B\big(\nabla^2 \mathcal{L}(u)\big)^\dagger \nabla \mathcal{L}(u)$ yields

$$B\epsilon^+ = B\big(\nabla^2 \mathcal{L}(u)\big)^\dagger \int_0^1 \big[\nabla^2 \mathcal{L}(u) - \nabla^2 \mathcal{L}(u^\star + t\epsilon)\big]\epsilon\, dt.$$

For each comparison, $m_e(u^\star + t\epsilon) = m_e^\star + t\Delta_e$ and $b_e^\top \epsilon = \Delta_e/\beta$, and, for $\Delta_e \neq 0$, $\int_0^1 \omega(m_e^\star + t\Delta_e)\, dt = \big(\sigma(m_e) - \sigma(m_e^\star)\big)/\Delta_e$. Hence, the integral equals $\beta\, B^\top \chi$, where $\chi \in \mathbb{R}^{|\mathcal{E}|}$ with

$$\chi_e = n_e\big[\omega(m_e)\,\Delta_e - \big(\sigma(m_e) - \sigma(m_e^\star)\big)\big] = \tfrac{1}{2}\, n_e\, \omega'(m_e^\star)\,\Delta_e^2 + O(\Delta_e^3).$$

Substituting $\omega' = \omega(1 - 2\sigma)$, the leading term of χ_e becomes $\frac{1}{2}\, n_e\, \omega(m_e^\star)\,\psi_e$ with $\psi \in \mathbb{R}^{|\mathcal{E}|}$ the vector of entries $\psi_e = \big(1 - 2\sigma(m_e^\star)\big)\Delta_e^2$, that is, $\chi = \frac{1}{2} D^\star \psi + O(\|\epsilon\|^3)$.

In terms of χ, the expression of $B\epsilon^+$ reads $B\epsilon^+ = B\big(\nabla^2 \mathcal{L}(u)\big)^\dagger \beta B^\top \chi$, where $\beta B^\top \chi = O(\|\epsilon\|^2)$. Lemma D.1 with the pair $(u^\star, u)$ gives $\mathrm{e}^{-\tau}\nabla^2 \mathcal{L}(u^\star) \preceq \nabla^2 \mathcal{L}(u) \preceq \mathrm{e}^{\tau}\nabla^2 \mathcal{L}(u^\star)$ with $\tau =$

$\max_e |\Delta_e| = O(\|\epsilon\|)$; inverting the Hessians on $\mathcal{H}$ yields $\big(\nabla^2\mathcal{L}(u)\big)^\dagger = \big(\nabla^2\mathcal{L}(u^\star)\big)^\dagger + O(\|\epsilon\|)$. Therefore, we let $\mathsf{Q} := B(B^\top D^\star B)^\dagger B^\top D^\star$ and obtain

$$B\epsilon^+ = B\,\beta^{-2}(B^\top D^\star B)^\dagger\,\beta B^\top\big(\tfrac{1}{2}D^\star\psi\big) + O(\|\epsilon\|^3) = \tfrac{1}{2\beta}\,\mathsf{Q}\psi + O(\|\epsilon\|^3),$$

With $\|a\|_{D^\star} = (a^\top D^\star a)^{1/2}$ for $a \in \mathbb{R}^{|\mathcal{E}|}$, the factorization $\nabla^2\mathcal{L}(u^\star) = \beta^2 B^\top D^\star B$ gives $\|\epsilon^+\|_{\nabla^2\mathcal{L}(u^\star)} = \beta\,\|B\epsilon^+\|_{D^\star}$, and hence

$$\big\|\epsilon^+\big\|_{\nabla^2\mathcal{L}(u^\star)} = \tfrac{1}{2}\,\|\mathsf{Q}\psi\|_{D^\star} + O(\|\epsilon\|^3). \tag{26}$$

It remains to bound $\|\mathsf{Q}\psi\|_{D^\star}$. The matrix $B^\top D^\star B = \beta^{-2}\nabla^2\mathcal{L}(u^\star)$ also has the kernel $\mathrm{span}\{\mathbf{1}\}$ by Proposition 4.2, and the same argument gives $(B^\top D^\star B)^\dagger B^\top D^\star B = I - \mathbf{1}\mathbf{1}^\top/K$, which, together with $B\mathbf{1} = 0$, gives $\mathsf{Q}B = B$ and $\mathsf{Q}^2 = \mathsf{Q}$. In addition, $D^\star\mathsf{Q} = D^\star B(B^\top D^\star B)^\dagger B^\top D^\star$ is symmetric, which means that Q is self-adjoint with respect to the inner product $\langle a, b\rangle_{D^\star} = a^\top D^\star b$, and hence Q is the $D^\star$-orthogonal projection onto $\mathrm{range}(B)$, which gives $\|\mathsf{Q}\psi\|_{D^\star} \le \|\psi\|_{D^\star}$. On the other hand, $\|\psi\|_{D^\star}$ can be bounded as follows,

$$\|\psi\|_{D^\star}^2 = \sum_e n_e\omega(m_e^\star)\big(1-2\sigma(m_e^\star)\big)^2\Delta_e^4 \le \Big(\max_e|1-2\sigma(m_e^\star)|\,\max_e|\Delta_e|\Big)^2\sum_e n_e\omega(m_e^\star)\Delta_e^2,$$

where $\sum_e n_e\omega(m_e^\star)\Delta_e^2 = \epsilon^\top\nabla^2\mathcal{L}(u^\star)\epsilon = \|\epsilon\|^2_{\nabla^2\mathcal{L}(u^\star)}$. Recalling the expression (26) yields

$$\big\|\epsilon^+\big\|_{\nabla^2\mathcal{L}(u^\star)} \le \tfrac{1}{2}\max_e|1-2\sigma(m_e^\star)|\,\max_e|\Delta_e|\;\|\epsilon\|_{\nabla^2\mathcal{L}(u^\star)} + O(\|\epsilon\|^3),$$

which is the bound (17) with $c_\star = \frac{1}{2}\max_e|1-2\sigma(m_e^\star)| = \max_e\big|\sigma(m_e^\star)-\frac{1}{2}\big|$. As σ takes values in $(0,1)$, we have $c_\star \le \frac{1}{2}$; moreover, it is at most $\sigma(1) - \frac{1}{2} = \frac{\mathrm{e}-1}{2(\mathrm{e}+1)}$ when $\beta\max_{(w,l)\in\mathcal{E}}|u_w^\star - u_l^\star| \le 1$. □

D.3 The sampled direction as an inexact Newton step

This subsection proves Proposition 6.2, with $\widehat{d}(u)$, $\overline{\xi}$ and ζ as defined before it in Section 6.

Proof of Proposition 6.2. Abbreviate $A = \nabla^2\mathcal{L}(u)$, and write $\widehat{d}(u) = \frac{1}{M}\sum_{i=1}^M \widehat{d}_i$ with independent samples $\widehat{d}_i$ of the estimator (14), each satisfying $\mathbb{E}\,\widehat{d}_i = d(u)$. The error splits as $\widehat{d}(u) - \overline{\xi} = \big(\widehat{d}(u) - d(u)\big) + \big(d(u) - \overline{\xi}\big)$. For the sampling error $\widehat{d}(u) - d(u)$, the cross terms $\mathbb{E}\big[(\widehat{d}_i - d(u))^\top A(\widehat{d}_j - d(u))\big]$ with $i \ne j$ vanish by independence, and hence

$$\mathbb{E}\,\Big\|\widehat{d}(u) - d(u)\Big\|_A^2 = \frac{1}{M^2}\sum_{i=1}^M \mathbb{E}\,\Big\|\widehat{d}_i - d(u)\Big\|_A^2 = \frac{1}{M}\,\mathbb{E}\,\Big\|\widehat{d}_1 - d(u)\Big\|_A^2 = O(1/M),$$

where the last expectation is finite. The regularization bias $d(u) - \overline{\xi} = -\big[(\gamma I + A)^{-1} - A^\dagger\big]\nabla\mathcal{L}(u) = -\gamma(\gamma I + A)^{-1}\overline{\xi}$, where we use $\overline{\xi} = -A^\dagger\nabla\mathcal{L}(u)$. As $\overline{\xi}$ lies in the range of A, on which the eigenvalues of A are at least $\lambda_2(A)$, the A-norm of the regularization bias is at most $\frac{\gamma}{\gamma+\lambda_2(A)}\,\big\|\overline{\xi}\big\|_A$. The triangle inequality then gives the bound (18).

The convergence conclusions follow from the inexact Newton theorem of Dembo et al. (1982): relative residuals $\zeta^t \le c < 1$ in the local norm yield linear convergence, $\zeta^t \to 0$ yields superlinear convergence, and $\zeta^t = O(\|\nabla\mathcal{L}(u^t)\|)$ yields quadratic convergence, the last matching Theorem 6.1. □

E Acceleration via samplers under complete comparison graph

We briefly review the samplers proposed by Shi et al. (2025) to accelerate DPO, where a single prompt with K responses is fixed. They highlight the importance of the sampler to form a pair (y, y') of responses to be compared by sampling y from one distribution over the K responses and y' independently from a possibly different one. It is known that sampling both y and y' uniformly

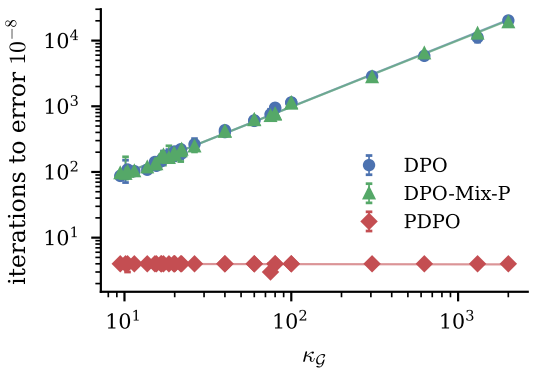


Figure 3: Iterations to reach 10^{-8} on the graphs with exactly $2K$ edges. The iteration counts of DPO and DPO-Mix-P grow as $\kappa_{\mathcal{G}}^{1.02}$ and $\kappa_{\mathcal{G}}^{1.01}$, respectively, whereas PDPO needs a median of 3 to 4 iterations.

yields a linear convergence rate for DPO. They proposed two new strategies. DPO-Mix-R mixes the uniform sampler with another one: y is sampled with probability proportional to $\mathrm{e}^{r(y)}$ and y' with probability proportional to $\mathrm{e}^{-r(y')}$, where r is the true reward. In addition, DPO-Mix-P replaces the true reward, which is unknown in practice, by $\beta \log(\pi_\theta/\pi_{\mathrm{ref}})$ with π_θ the current policy, and hence samples y and y' with probabilities proportional to $(\pi_\theta(y)/\pi_{\mathrm{ref}}(y))^\beta$ and $(\pi_{\mathrm{ref}}(y')/\pi_\theta(y'))^\beta$, respectively. Under both the strategies, the policy gradient descent method converges quadratically with the step size $1/(\beta^2 K)$. Note that the analysis relies on that every pair is compared, which means that the comparison graph is complete.

F Experimental details and additional results

F.1 Synthetic problems

We set $\beta = 3$ by default. The runs start from a centered Gaussian initialization with standard deviation 0.1; the sweeps run all six graph types at each $K \in \{20, 40, 60, 80, 100\}$ over five seeds. All runs use the rewards $r_y = (y-1)/(K-1)$ of Section 7.1. The weights $n_{(i,j)} = \sigma(r_i - r_j)$ and $n_{(j,i)} = \sigma(r_j - r_i)$ of a compared pair sum to one, and thus $u^\star = r/\beta$ is the exact minimizer.

For DPO and DPO-Mix-P, we tune the step size separately for each graph and seed over 61 logarithmically spaced values in $[10^{-3}, 10^2]$. We select the step size that reaches the target error in the fewest iterations; if no candidate reaches the target, the step size with the smallest final error is chosen. The target is 10^{-8} for Figure 2(c) and 10^{-15} for Figure 2(a, b) and Figure 4.

On the complete graph with $K = 20$, DPO-Mix-P and PDPO both reach an error below 10^{-15} in four iterations, which confirms the quadratic rate. Table 2 lists the iteration counts behind Figure 2(c) on the complete graph and on two path graphs.

In Figure 2(c), the number of comparisons changes with the graph type, from $K-1$ for the path and the star to $K(K-1)/2$ for the complete graph. To separate its effect from that of $\kappa_{\mathcal{G}}$, Figure 3 gives every graph the same number $2K$ of edges. With $2K$ edges, $\kappa_{\mathcal{G}}$ ranges from 9.4 to 2000 over the graph types and sizes, and the iteration counts of DPO and DPO-Mix-P grow with $\kappa_{\mathcal{G}}$, whereas those of PDPO do not.

F.2 Language-model experiments

The experiments fine-tune Qwen2.5-1.5B-Instruct (Yang et al., 2024) with preference labels on the responses to 256 prompts and compare DPO with PDPO in two settings: in the offline setting, all comparisons of a comparison graph on the responses of each prompt are labeled, and in the online setting, only one pair of responses per prompt is labeled.

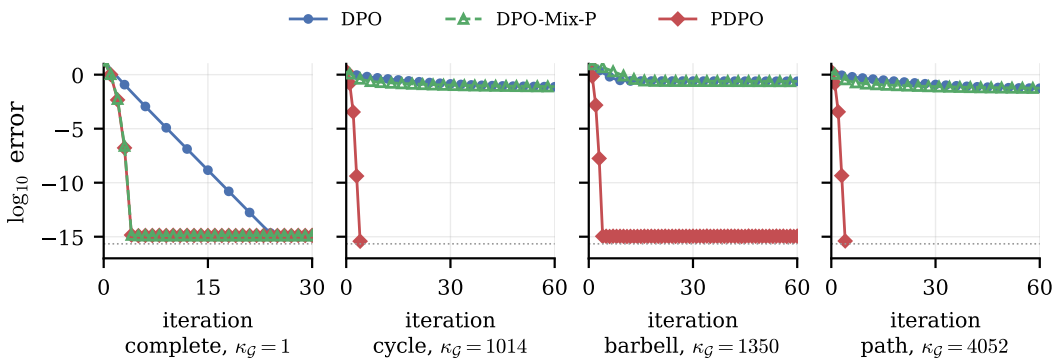


Figure 4: Error against iteration on four comparison graphs with $K = 100$.

Table 2: Iterations to reach $\|u-u^\star\|_{\nabla^2\mathcal{L}(u^\star)} < 10^{-8}$ on the complete graph with $K = 20$ ($\kappa_\mathcal{G} = 1.0$) and on paths with $K = 20$ ($\kappa_\mathcal{G} = 161$) and $K = 100$ ($\kappa_\mathcal{G} = 4.1 \cdot 10^3$), averaged over five seeds.

Method	Complete, $K = 20$	Path, $K = 20$	Path, $K = 100$
DPO	10	1 352	32 108
DPO-Mix-P	4	1 569	37 258
PDPO	4	4	4

Model. The policy π_θ is initialized at Qwen2.5-1.5B-Instruct, the same as the frozen reference policy π_{ref}. For a prompt x and a response y, the score is $u_\theta(y) = \log \pi_\theta(y \mid x) - \log \pi_{\text{ref}}(y \mid x)$, where x is formatted by the chat template of the model and the log-probabilities are summed over the tokens of y; hence $u_\theta = 0$ at the initialization. Both DPO and PDPO use $\beta = 0.1$ in objective (5).

Training protocol. The prompts come from the `train_prefs` split of `HuggingFaceH4/ultrafeedback_binarized` (Cui et al., 2024). Following the data construction of Meng et al. (2024), the initial policy Qwen2.5-1.5B-Instruct samples, before training, five responses to each of 20000 prompts drawn at random from this split, at temperature 0.8 with nucleus sampling at threshold 0.95 and at most 4096 new tokens, and the reward model ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024) assigns a reward to each full response. The 256 training prompts are drawn at random from these prompts, after excluding the prompts with more than 1800 tokens under the chat template and the prompts with five identical responses. Identical responses of a prompt are merged, and each prompt–response sequence is truncated to 2048 tokens. Then 13, 10, 6 and 227 prompts have 2, 3, 4 and 5 distinct responses, respectively.

In the offline setting, the responses of each prompt are connected by a path, a complete graph, a cycle or a star, and every connected pair is labeled by the ArmoRM rewards: the response with the larger reward is preferred, and equal rewards give a tie. DPO on these comparisons and PDPO with the direction of Algorithm 2 are trained with AdamW at the learning rates 8×10^{-6} and 2×10^{-5}, respectively. In the online setting, only one pair of responses per prompt, drawn uniformly at random, is labeled in the same way. DPO is trained on this pair with AdamW at the learning rate 4×10^{-6}, whereas PDPO takes the learning rate 2×10^{-5} and estimates the preconditioned direction by random walks according to Algorithm 4. We note that the adopted learning rates are selected from a grid by the ranking accuracy on the validation set.

In both settings, each method takes 64 updates on minibatches of four prompts, one pass over the 256 prompts, and the loss of a minibatch is the average of the losses of its prompts. Training runs in BF16, including the optimizer states, on one GPU with 80 GB of memory, and the parameters after the last update are evaluated.

Evaluation. The 4096 evaluation prompts and the 512 prompts of the validation set, used for the learning rate selection above, are two disjoint random sets of `train_prefs` prompts with at most

160 tokens under the chat template, outside the pool of 20000 prompts and the prompts of earlier experiments. For each of them, Qwen2.5-0.5B-Instruct (Yang et al., 2024) samples two responses y and y' at temperature 1 with at most 512 new tokens, and the reward model Skywork-Reward-V2-Qwen3-1.7B (Liu et al., 2026) assigns them the rewards r and r'; neither model is used for the training data. The *ranking accuracy* of a policy is the fraction of pairs in which $u_\theta(y) - u_\theta(y')$ and $r - r'$ have the same sign, where a pair with $u_\theta(y) = u_\theta(y')$ or $r = r'$ counts as one half.

The *win rate* compares the answers generated by PDPO and DPO trained on the training set. We sample 512 prompts randomly from the prompts of AlpacaEval (Li et al., 2023), and none of them appears among the training prompts. Each model answers every prompt with vLLM in BF16 at temperature 0.7, without nucleus sampling or repetition penalty, and with at most 2048 new tokens. The judge GPT-5.5, receives the prompt and the two answers without the method names and returns the better answer or a tie; it is instructed to weigh correctness, relevant completeness, instruction following, clarity and safety. With W and T the numbers of wins and ties of PDPO, the win rate is defined by $100(W + T/2)/512$.