# Robust Multi-Objective Controlled Decoding of Large Language Models

**Seongho Son** [*1] **William Bankes** [*1] **Sangwoong Yoon** [*2]
**Shyam Sundhar Ramesh** [*2] **Xiaohang Tang** [2] **Ilija Bogunovic** [2 3]

## Abstract

We introduce Robust Multi-Objective Decoding (RMOD), a novel inference-time algorithm that robustly aligns Large Language Models (LLMs) to multiple human objectives (e.g., instruction-following, helpfulness, safety) by maximizing the worst-case rewards. RMOD formulates the robust decoding problem as a *maximin* two-player game between adversarially computed reward weights and the sampling policy, solvable through a Nash equilibrium. We demonstrate that this game reduces to a convex optimization problem to identify the worst-case reward weights, with the optimal sampling policy analytically derived. For practical applications, we propose an efficient algorithm of RMOD tailored for contemporary LLMs, introducing minimal computational overhead compared to standard non-robust Controlled Decoding methods. Experimental results across the range of popular alignment datasets with up to 10 objectives show the effectiveness of RMOD and its distilled version, consistently outperforming baselines in worst-case rewards and win rates.

## 1. Introduction

Large Language Models (LLMs) require alignment to become useful and safe conversational agents (Rafailov et al., 2023; Azar et al., 2023; Hong et al., 2024; Ethayarajh et al., 2024; Wu et al., 2024). Recent approaches frame alignment as a multi-objective problem (Zhao et al., 2023; Shi et al., 2024) that aims to balance various objectives simultaneously, e.g., *helpfulness*, *truthfulness*, *honesty*, and *safety* in the resulting model (Bai et al., 2022; Cui et al., 2023; Sorensen et al., 2024). Inference-time alignment algorithms (Shi et al., 2024; Wang et al., 2024b; Dong et al., 2023; Rame et al., 2024) such as Controlled Decoding (Mudgal et al., 2023) have become popular, as they enable reweighing different objectives at deployment without expensive retraining.

However, multi-objective alignment naturally poses an important question: *How can we balance multiple diverse and often competing objectives at inference time?* When working with critical objectives, it is important that none of them drops below a certain level. For example, consider when safety is one of the objectives: its importance should never be neglected during the response generation while we also do not want overly cautious responses that refuse to provide any information at all. This motivates a *robust* approach, which finds a policy that maximizes the least aligned combination of objectives (Yoon et al.; Ramesh et al., 2024; Chakraborty et al., 2024). Previous work has focused on finding algorithms with good Pareto frontiers (Shi et al., 2024; Rame et al., 2024), rather than a practical approach to find a robust weighting of the objectives at inference time.

In order to address this question, we introduce **Robust Multi-Objective Decoding** (RMOD), a novel *test-time* alignment algorithm designed to generate *robust* responses by *maximizing alignment to the worst-case weightings over the objectives*. Using the value functions trained for each objective, RMOD dynamically reweights alignment objectives during decoding to improve the least aligned objective; see Figure 1. Our main contributions are as follows: **(i)** We propose an algorithm that achieves balanced alignment without requiring any information about the relative importance of the objectives; **(ii)** We present the algorithm of RMOD that performs blockwise best-of-$K$ w.r.t. the worst-case weighted sum of values, incurring minimal compute overhead; **(iii)** We rigorously evaluate RMOD on diverse multi-objective datasets, demonstrating the effectiveness of our method in robust alignment. Our results demonstrate that RMOD achieves higher worst-case rewards than baselines, while showing a win rate of over 80% in the Helpful-Harmless (Bai et al., 2022) dataset against the reference policy.

[*]Equal contribution [1]Department of Computer Science, University College London, London, United Kingdom [2]Department of Electric Engineering, University College London, London, United Kingdom [3]Centre for Artificial Intelligence, University College London, London, United Kingdom. Correspondence to: Seongho Son <seong.son.22@ucl.ac.uk>.

*ICML 2025 Workshop on Models of Human Feedback for AI Alignment*, Vancouver, Canada. 2025. 

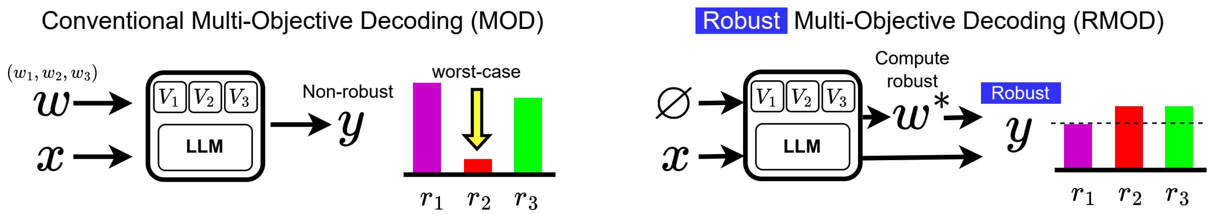



*Figure 1.* (Left) Existing multi-objective alignment methods require the weights for each reward. (Right) RMOD produces a robust response $y$ when a prompt $x$ is given, using the value functions for each objective $V_i$ and the worst-case weights $w^*$.

## 2. Related Work

Multi-objective alignment of policies (Li et al., 2020) is an important area of research in Reinforcement Learning (RL). Optimizing for multiple objectives is also essential to correctly align LLMs (Vamplew et al., 2018), as modern applications demand a range of alignment goals (Wang et al., 2024a; 2023; Chen et al., 2024). A common approach to aligning models with multiple objectives is to weight different alignment objectives at training (Zhou et al., 2023; Dong et al., 2023) or inference time (Shi et al., 2024; Wang et al., 2024b; Dong et al., 2023; Rame et al., 2024). The weights on these objectives can be provided as a context (Shi et al., 2024; Wang et al., 2024b; Dong et al., 2023) to the model; used to combine the weights of a diverse set of models (Rame et al., 2024; Feng et al., 2024; Jang et al., 2023); or are included within the prompt itself (Wang et al., 2024a; Castricato et al., 2024). These weights are a key component in ensuring the correct model alignment but are often not known in practice.

To address this, (Shi et al., 2024) propose finding weightings using a hyperparameter search across a validation set. Other works suggest merging model parameters (Mavromatis et al., 2024) or implicitly weighting objectives using the learned contexts (Zhao et al., 2023). However, (Hwang et al., 2023; Li et al., 2023) show that demographic information is not necessarily predictive of the correct alignment of individuals. Finally, (Poddar et al., 2024; Chen et al., 2024) leverage previous interactions to learn a model that predicts suitable weights across attributes. All these directions require additional information at inference time, be it about the users themselves or examples of prior interactions.

Although other works (Ramesh et al., 2024; Chakraborty et al., 2024; Maura-Rivero et al., 2025) have considered robust alignment over a group of attributes, we are the first to consider such an objective in the inference-time alignment setting. Inference time approaches offer more flexibility than fine-tuning methods, as their alignment can be easily changed without further retraining (Mudgal et al., 2023; Zhou et al., 2024; Kong et al., 2024; Khanov et al., 2024). They also offer further performance improvements by scaling test time compute (Snell et al., 2024). We detail additional related works in Appendix E.

## 3. Problem Formulation

Let $\pi_{\text{ref}}(\cdot)$ denote a reference language model that generates a response $y$ for a given prompt $x \in \mathcal{X}$, where $\mathcal{X}$ is the set of prompts. The response $y = [y_1, \ldots y_T]$ consists of $T$ tokens where each token $y_t$ is drawn from the token vocabulary $\mathcal{Y}$. We denote the probability of response $y$ given the prompt $x$ as $\pi_{\text{ref}}(y|x)$. We aim to adapt responses $y$ sampled from $\pi_{\text{ref}}(\cdot|x)$ to align with *multiple objectives at inference time*. Specifically, we define our objectives through reward models that evaluate the desirability of a response w.r.t. various attributes (e.g., conciseness, harmlessness, accuracy, etc.). We denote the objectives as $g \in \mathcal{G}$, where $|\mathcal{G}| = G$, and the reward models as $\mathcal{R}_G = \{R_g(x, y)\}_{g=1}^G$ corresponding to $G$ objectives. Here, $R_g(x, y)$ is a scalar function embodying objective $g$ that evaluates how desirable the response $y$ given the prompt $x$ is. Following standard practices in the literature (Mudgal et al., 2023; Dai et al., 2023; Ouyang et al., 2022), we define a token-wise reward $R_g$ as

$$R_g(x, y^t) = \begin{cases} r_g(x, y^t) & \text{if} \quad y_t = \text{EOS}, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Here, $y^t = [y_1, \ldots, y_t]$ denotes a subsequence of $t$ tokens, where each token $y_t$ is drawn from the token vocabulary $\mathcal{Y}$. We use the reward function $r_g(\cdot, \cdot)$ to evaluate the reward of the final response $y$. The alignment of response $y$ to the $G$ objectives is measured through the weighted multi-objective reward $\sum_{g=1}^G w_g R_g(x, y)$. Here, $w = (w_1, \ldots, w_G)$, and $\Delta^{G-1}$ represent weights over the $(G-1)$-dimensional simplex and express the significance over the reward objectives.

We consider a *block-wise decoding* procedure (as in (Mudgal et al., 2023)) for generating the response $y$. In essence, at each decoding step $t$ given prompt $x$ and partially decoded response $y^t$, we seek a robust policy $\pi^*(\cdot|x, y^t)$, that is aligned to the worst-case weightings over the $G$ objectives and provides probabilities over the set $\mathcal{Z} = \mathcal{Y}^B$ for the next sequence block $z$ consisting of $B$ tokens. As $y_t$ also denotes a block with $B$ tokens in this case, $R_g(x, y^t) = r_g(x, y^t)$ if EOS $\in y_t$ and $R_g(x, y^t) = 0$ otherwise in the block-wise setting. We formalize this objective later in this section.

**Value Function.** We formalize the robust objective for policy $\pi(\cdot|x, y^t)$ at each decoding step $t$ using *value* functions $V_g$ for $g \in \mathcal{G}$. This is for measuring the alignment of the expected response towards the $G$ objectives at each step $t$. Given $\pi_{\text{ref}}$ and the reward $R_g(\cdot)$ corresponding to objective $g$, the value of a partial sequence $y^t$ is the expected reward attained by following $\pi_{\text{ref}}$ and expressed as:

$$V_g\left(x, y^t\right) := \mathbb{E}_{z_1, z_2, \ldots \sim \pi_{\text{ref}}}\Big\{\sum_{\tau \geq 1} R_g\left(x, [y^{t+\tau-1}, z_\tau]\right)\Big\}, \tag{2}$$

where, $z_\tau \sim \pi_{\text{ref}}(\cdot|x, y^{t+\tau-1})$ and $y^{t+\tau} = [y^{t+\tau-1}, z_\tau]$. We denote the value of choosing a particular sequence $z$ at the next step $t+1$ and following $\pi_{\text{ref}}$ afterward as $V_g(x, y^t; z)$. Moreover, we define the value function of a given policy $\pi$ as the expected value after sampling $z$ at the next step $t+1$ from $\pi$:

$$V_g\left(x, y^t; \pi\right) = \mathbb{E}_{z \sim \pi}[V_g\left(x, y^t; z\right)]. \tag{3}$$

**Robust Objective.** We describe the objective for a robust policy as a *max-min game* at each decoding step $t$ in terms of $V_g(x, y^t; \pi)$,

$$\max_{\pi} \min_{w \in \Delta^{G-1}} \lambda \sum_{g=1}^{G} w_g V_g(x, y^t; \pi) - D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}). \tag{4}$$

Here, the value function $V_g$ quantifies the impact of selecting a specific sequence $z$ at decoding step $t$ on the expected reward $R_g(x, y^T)$ of the fully decoded response $y^T$. We regularize this objective with the KL divergence to ensure that the response remains probable under the reference policy $\pi_{\text{ref}}$ w.r.t. a trade-off parameter $\lambda$. Moreover, the above optimization problem is a two-player zero-sum game, where the policy $\pi$ and weights $w$ act as opponents with inversely related payoffs. The policy $\pi$ and the weights $w$ represent stochastic (mixed) strategies, modeled as categorical distributions of choosing sequence $z$ and group $g$, respectively.

## 4. Robust Multi-Objective Decoding

In this section, we discuss our proposed algorithm for solving the robust objective in Equation (4). We show how RMOD obtains the optimal weights and policy at a Nash Equilibrium, and also discuss the properties of the weights at convergence. We also present an alternative approach to the max-min game in Appendix D, which is no-regret learning.

**Minimax Reformulation.** The objective in Equation (4) is clearly linear in $w$. Moreover, it is concave in $\pi$ because the value function $V_g(x, y^t; \pi)$ is linear in $\pi$, and the KL-divergence $D_{\text{KL}}(\pi \parallel \pi_{\text{ref}})$ is convex in $\pi$. We assume that the space of $\pi(\cdot|x, y^t)$ is a convex class of probability measures. Hence, as the set of strategies for both players ($\pi$ and $w$) are compact and correspond to mixed strategies, the existence of a Nash Equilibrium (NE) for Equation (4) is guaranteed due to Nash's existence theorem (Nash Jr, 1950). Because the objective in Equation (4) is concave-convex in terms of $\pi$ and $w$, the minimax theorem (v. Neumann, 1928; Sion, 1958) allows the interchange minimum and maximum operators in the objective. Thus, for each decoding step $t$ we can re-write the robust objective as

$$\min_{w \in \Delta^{G-1}} \max_{\pi} \lambda \sum_{g=1}^{G} w_g V_g(x, y^t; \pi) - D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}). \tag{5}$$

Note that the inner maximization in Equation (5) is in line with the standard KL-regularized RLHF objective. Here, $\lambda > 0$ trades off the weighted value of policy $\pi$ for a deviation of $\pi$ from the reference model $\pi_{\text{ref}}$. Moreover, due to the strict convexity of $D_{\text{KL}}(\pi \parallel \pi_{\text{ref}})$ w.r.t. $\pi$ for a fixed $\pi_{\text{ref}}$, the maximization problem is strictly concave. Consequently, the optimal policy for the inner maximization problem is unique for any given weights $w$ and trade-off parameter $\lambda$, and we characterize the policy in the following proposition.

**Proposition 4.1.** *Given the value functions $V_g$ for each objective $g \in \mathcal{G}$, the solution to the inner maximization problem in Equation (5) is unique for any given weights $w$, normalization constant $Z(x, y^t, w)$, and trade-off parameter $\lambda$, and can be expressed as*

$$\pi(z|[x, y^t]; w) = \frac{\pi_{\text{ref}}(z|[x, y^t]) \exp\left(\lambda \sum_{g=1}^{G} w_g V_g(x, y^t; z)\right)}{Z(x, y^t, w)}. \tag{6}$$

**Algorithm 1** RMOD Algorithm

**Input:** Prompt $x$, learnt value functions $\{V_g(\cdot; \theta)\}_{g \in \mathcal{G}}$, reference policy $\pi_{\text{ref}}$, action space $\mathcal{Z}$, regularisation coefficient $\lambda > 0$, number of candidates $K$, block size $B$, weight update iteration limit $I$
$y^0 = \emptyset$
**for** $t \in [T]$ **do**
  $z_{(k)} \sim \pi_{\text{ref}}(\cdot|[x, y^t])\ \forall k \in [K]$
  // Sample $K$ blocks of length $B$
  $V_g(x, y^t; z_{(k)}, \theta)$ for all $g \in \mathcal{G},\ k \in [K]$
  // Calculate values of blocks
  Update weights (Equation (10)) $I$ times:
  // Iteratively solve for weights
  $w_{g,i+1} = w_{g,i} \cdot \exp\Big[-\eta \sum_{k=1}^{K} \pi_{\text{ref}}(z_k \mid [x, y^t]) h(z_k; x, y^t, w_i, g)\Big]$
  $y_{t+1} = \arg\max_{z_{(k)}} \sum_{g=1}^{G} w_{g,I} \cdot V_g\left(x, y^t; z_{(k)}, \theta\right)$
  // Choose block
  $y^{t+1} = [y^t, y_{t+1}]$
  // Append the selected block
**end for**
Return $y^T$

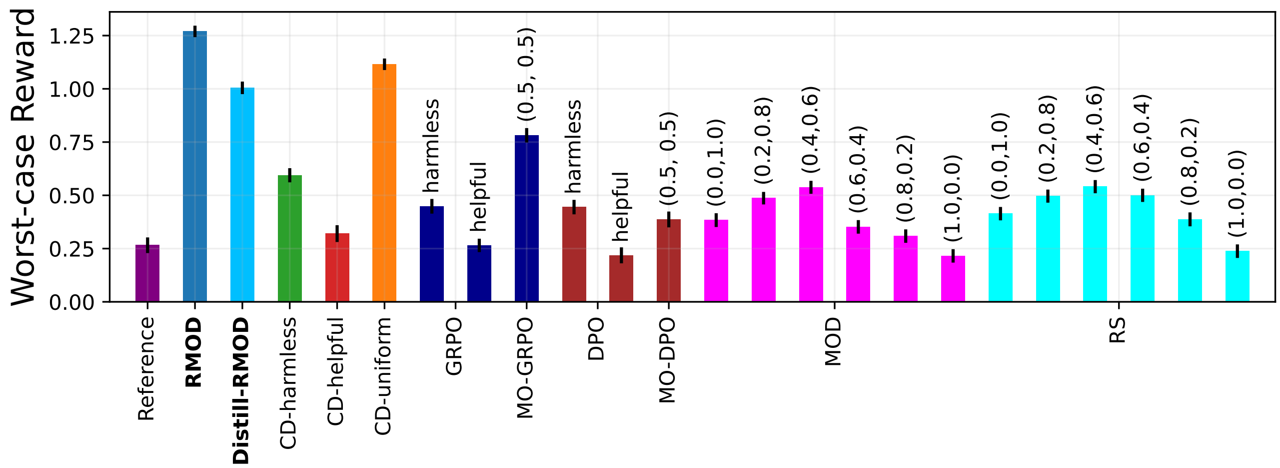



*Figure 2.* Worst-case reward obtained by RMOD and baselines in the **HH dataset**. We use $B = 16$, $K = 16$ for all the decoding methods and $\lambda = 0.5$ for RMOD. For every method, we use `gemma-2-2b-it` for the base model. Texts at the top of the bars indicate the chosen objective or weights for each objective. Methods having the prefix CD- denote the controlled decoding baselines. RS and MOD use the models trained with GRPO. RMOD achieves the best worst-case reward and outperform all the baselines, while its distilled variant DISTILL-RMOD shows higher worst-case reward than all the non-controlled decoding baselines.

Here, the weights-conditioned policy, $\pi(\cdot|\cdot; w)$, is the *best-response policy* to weights $w$. We defer the proof of this proposition to Appendix A.1. Proposition 4.1 establishes that given a set of weights $w$, the reference policy $\pi_{\text{ref}}$, and the value functions $V_g$, one can employ Equation (6) to sample from a policy that aligns with the objectives while staying close to the reference policy in terms of KL divergence. Moreover, it enables us to develop an inference-time alignment method that keeps the reference model frozen while combining its logits with the value functions $V_g$ to achieve the alignment objective.

Plugging Equation (6) back to Equation (5), we obtain the following simplified optimization problem with respect to $w$ (derivation is provided in Appendix A.3):

$$w^* = \underset{w \in \Delta^{G-1}}{\arg\min} \log \mathbb{E}_{z \sim \pi_{\text{ref}}(\cdot|[x,y^t])} \Big[ f(z; x, y^t, w) \Big], \quad \text{and} \quad f(z; x, y^t, w) = \exp\left( \sum_{g=1}^{G} \lambda w_g V_g(x, y^t; z) \right). \tag{7}$$

Here, $w^*$ is the NE solution of Equation (4). We obtain the corresponding *best-response policy* $\pi^* = \pi(\cdot|\cdot; w^*)$ by substituting $w^*$ in Equation (6). We formally detail this in the following proposition.

**Proposition 4.2.** *The solution $w^*$ to the convex optimization problem in Equation* (7) *and $\pi^* = \pi(\cdot|\cdot; w^*)$ in Equation* (6) *constitute a Nash Equilibrium for the max-min game in Equation* (4).

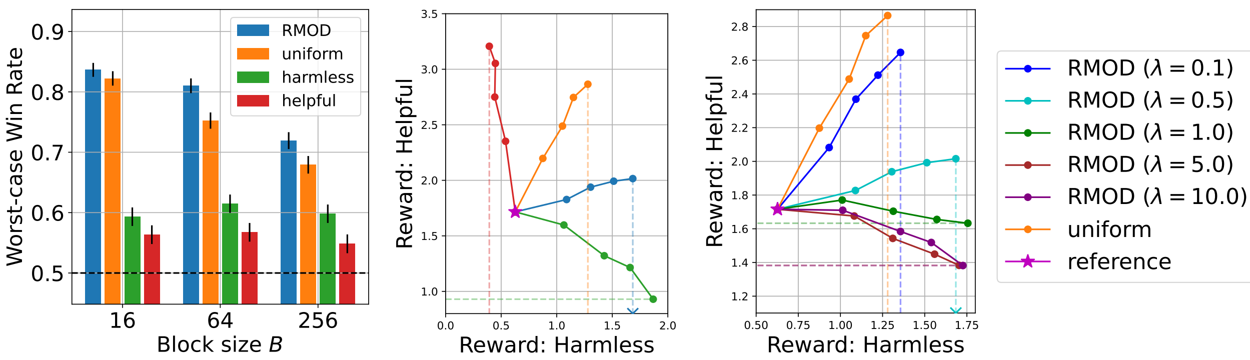
In contrast to the initial objective presented in Equation (4), Equation (7) represents a non-linear optimization problem solely in terms of the variable $w$. Notably, Equation (7) constitutes a convex optimization problem by including the *LogSumExp* function, which is convex (El Ghaoui, 2017). This convexity guarantees the existence of a global minimum, which can be identified through the search for a local minimum. Furthermore, the dimensionality of $w$ is generally smaller than that of the space defined by $\pi$, making Equation (7) amenable to solve by using iterative techniques such as gradient descent, which efficiently approximates the optimal solution. We note that the evaluation of $\pi_{\text{ref}}(z|[x, y^t])$ and $V_g(x, y^t; z)$ is performed only once as $\pi_{\text{ref}}(z|[x, y^t])$ and $V_g(x, y^t; z)$ are independent of $w$. Hence, in order to solve Equation (7), we propose running an iterative algorithm based on the inferred values $V_g$ to find the worst-case weights $w^*$ that minimize the exponential of the weighted values.

The RMOD algorithm is designed to yield a robust policy at each decoding step. However, in practical applications, minimizing the latency of RMOD remains critical. In Section 5, we introduce components designed to mitigate the high-latency challenges associated with the RMOD algorithm.

**Behavior of Optimal Weights in RMOD.** We analyze Equation (7) using the KKT conditions in Appendix B to study the behaviour of $w^*$. We show that the weights $w_g^*$ equalize the expected future rewards across objectives, leading to robust alignment. The value of $\lambda$ determines the sparsity of $w^*$. Low values of $\lambda$ result in high entropy across the weights, while high values of $\lambda$ lead to low entropy with the majority of weight applied to a single objective.

## 5. Practical Implementation of RMOD

This section introduces RMOD (Algorithm 1), a low-latency inference-time alignment algorithm that outputs




(a) (b) (c)

*Figure 3.* Comparative study on the **HH dataset** between different decoding methods. In Figure 3a, we present the worst-case win rates against the reference policy across block sizes $B \in \{16, 64, 256\}$. As $B$ decreases from 256 to 16, the worst-case win rate of RMOD increases, consistently outperforming the baselines. Figure 3b shows the rewards obtained with $B = 16$ with different $K$, while using the same legend as Figure 3a. The purple star represents the average reward of $\pi_{\text{ref}}$, and the dots represent increasing K values (2, 4, 8, 16) as they move away from the purple star. RMOD improves the worst-case reward, having higher harmlessness reward than UNIFORM. (3c) Testing different values of $\lambda$ for RMOD with $B = 16$. We demonstrate that as $\lambda$ increases, RMOD concentrates on improving the worst-case reward. Doing the opposite makes RMOD more similar to UNIFORM decoding.

a robust response $y^T$ of length $T$ given a prompt $x$. In particular, RMOD is characterized by the following attributes: (i) It requires, as input, value functions $V_g$ trained via reward models $R_g$, (ii) It approximates the evaluation of Equation (7), computing $\hat{w}^*$ (Line 4-6 of Algorithm 1), using $K$ samples from the reference policy, (iii) Based on the computed weight and values of each sample, it approximates the robust policy $\pi^*(\cdot|[x, y^t])$ by selecting one of the samples (Line 7 of Algorithm 1). Finally, it concatenates the selected sequence to the previously decoded subsequence, and enters the next decoding step $t + 1$ (Line 8, 3 of Algorithm 1). We discuss the details of each attribute below.

### 5.1. Training the Value Functions

Note that RMOD requires evaluations from value functions $V_g$, whereas we only have the reward models corresponding to the $G$ objectives. Therefore, our goal is to train $G$ value functions that approximate $V_g(\cdot, \cdot)$ for each $g \in \mathcal{G}$. Since true $V_g$ are unavailable, we follow the **CD-FUDGE** (Yang & Klein, 2021) approach to train the value functions with parameters $\theta$ using the rewards of the final response $r_g(x, y)$:

$$\mathbb{E}_{x \sim \mu, y \sim \pi_{\text{ref}}(\cdot|x)} \sum_{1 \le t \le |y|} \big(V_g(x, y^t; \theta) - r_g(x, y)\big)^2 . \quad (8)$$

We discuss further details regarding the training of value functions for the experiments in Section 6.

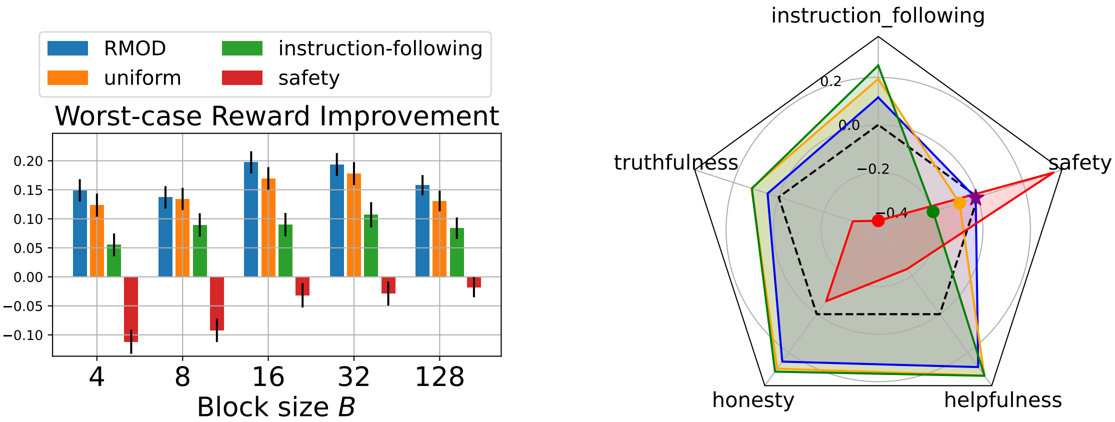
### 5.2. Block-wise RMOD

The length of the sequence $z$ plays a crucial role in the computation cost and alignment performance of the decoding algorithm. The number of decoding steps $T$, executed by Algorithm 1 for a given prompt $x$, reduces as the length of $z$ increases. When $z$ corresponds to a single token, the decoding process simplifies to token-wise decoding. However, this method requires computing the values for all samples, $\{z_k\}_{k=1}^K$, at each token, resulting in high computational costs (see Line-5 of Algorithm 1). To address this limitation, we adapt the RMOD algorithm to incorporate blockwise decoding (Mudgal et al., 2023), as detailed in Algorithm 1. In this formulation, $z$ represents a block of $B$ tokens, where $B$ can range from one to the maximum token length for each response. Notably, when each block constitutes a complete response, blockwise RMOD corresponds to a robust version of Best-of-$K$ rejection sampling (Stiennon et al., 2020; Nakano et al., 2021; Touvron et al., 2023), wherein the response with the maximum weighted-average value is selected. In Algorithm 1, at each step $t$, an entire block of $B$ tokens is selected from $K$ generated candidates. This modification significantly reduces the required number of value function evaluations compared to token-wise decoding, thereby enhancing the scalability of our algorithm.

### 5.3. Approximate Computation of Optimal Weights

The value function proposed in Section 5.1 predicts $V_g(x, y^t; z)$ for each $z$ individually. Consequently, one needs to perform $|\mathcal{Z}|$ forward passes through the trained value function to evaluate the expectation over all possible sequences $z \in \mathcal{Z}$ in Equation (7). We note that in practical settings $|\mathcal{Z}|$ is large and when $|z| > 1$, i.e., a block of tokens or sentence (see Section 5.2), $|\mathcal{Z}|$ can grow exponentially.

We thus turn to approximate the expectation in the objective function in Equation (7) with a set of independent samples $\{z_k\}_{k=1}^K$, where $z_k \sim \pi_{\text{ref}}(\cdot|[x, y^t])$, $k = 1, \cdots, K$ (see Line-4 of Algorithm 1) and approximate the optimal




(a) (b)

*Figure 4.* Comparison of decoding algorithms in the **UltraFeedback** dataset. Figure 4a displays worst-case reward improvement from $\pi_{\text{ref}}$ for $B \in \{4, 8, 16, 32, 128\}$ and $K = 16$. RMOD improves worst-case reward over the reference policy the most and outperforms baselines at $B = \{4, 16, 32, 128\}$. Figure 4b displays average reward in the UltraFeedback dataset with $K = 16, B = 4$. The purple star denotes the worst-case reward of RMOD and corresponds to the SAFETY objective. UNIFORM decoding (orange) and INSTRUCTION-FOLLOWING (green) sacrifice SAFETY to improve the other objectives. RMOD successfully improves worst-case performance while minimizing trade-off in other objectives.

weight $w^*$ with $\hat{w}^*$. As discussed in Proposition 4.2, the approximated objective of Equation (7) is a convex optimization problem and therefore guaranteed to have a global minimizer. However, it is not possible to obtain a closed-form solution for the approximated objective directly. Hence, we propose using iterative methods such as projected gradient descent (GD) to attain the global minimizer. We note that due to the monotonically increasing nature of $\log$ function, the minimizer of the approximated objective of Equation (7) is the same as the minimizer of

$$\hat{w}^* = \arg\min_{w \in \Delta^{G-1}} \sum_{k=1}^{K} \pi_{\text{ref}}(z_k|[x, y^t]) f(z_k; x, y^t, w). \quad (9)$$

Further, we adopt a soft update by performing gradient descent w.r.t. the logits of the group weights, i.e., $\log w$. The corresponding update expression for $w$ is

$$\begin{aligned} w_{g,i+1} &:= w_{g,i} \\ &\cdot \exp\Big( -\eta \sum_{k=1}^{K} \pi_{\text{ref}}(z_k \mid [x, y^t]) h(z_k; x, y^t, w_i, g) \Big) \quad (10) \end{aligned}$$

for $h(z; x, y^t, w, g) = e^{\sum_{g=1}^{G} \lambda w_g V_g(x, y^t; z)} \lambda w_g V_g(x, y^t; z)$ (see Appendix A.4 for derivation). Hence, at each decoding step $t$, given $K$ independent samples $\{z_k\}_{k=1}^K$ from $\pi_{\text{ref}}(\cdot|[x, y^t])$, we initialize the weights as $w_0 = \{1/G, \cdots, 1/G\}$, and iteratively update it using Equation (10) (see Line-6 of Algorithm 1). This effectively approximates the solving of Equation (7) for practical settings.

### 5.4. Direct Sampling from Best Response Policy

Following $I$ iterations of weight updates as outlined in Line-6 of Algorithm 1, we obtain the robust policy by substituting the converged weights, $w = w_I$, back to Equation (6). However, exact computation of the best response policy $\pi(\cdot|[x, y^t]; w_I)$ is still expensive as one needs to calculate $\pi(z|[x, y^t]; w_I)$ for each $z$ individually, wherein the cardinality of $z \in |\mathcal{Z}|$ can be large. To mitigate this, we reuse the existing samples $\{z_k\}_{k=1}^K$ for efficiency and choose sample $z_k$ with the highest weighted average value, $\sum_{g=1}^{G} \lambda w_{g,I} V_g(x, y^t; z_k)$ (see Line-7 of Algorithm 1). This avoids additional evaluations using the reference model or the value function and reduces computational costs.

## 6. Experiments

In this section, we study the empirical performance of RMOD on various multi-objective datasets. Our code[1] is available online, and further details of the experiment setting and additional results are provided in Appendix C.

### 6.1. Experiment Settings

**Datasets.** We evaluate RMOD on the Anthropic Helpfulness-Harmless (HH) (Bai et al., 2022), UltraFeedback (Cui et al., 2023) and ValuePrism (Sorensen et al., 2024) datasets. We construct our training set for value function learning by generating 4 responses per prompt from $\pi_{\text{ref}}$ on the training split for the HH and ValuePrism datasets,

[1] Code available at: https://anonymous.4open.science/r/robust-multi-objective-decoding-98D5.

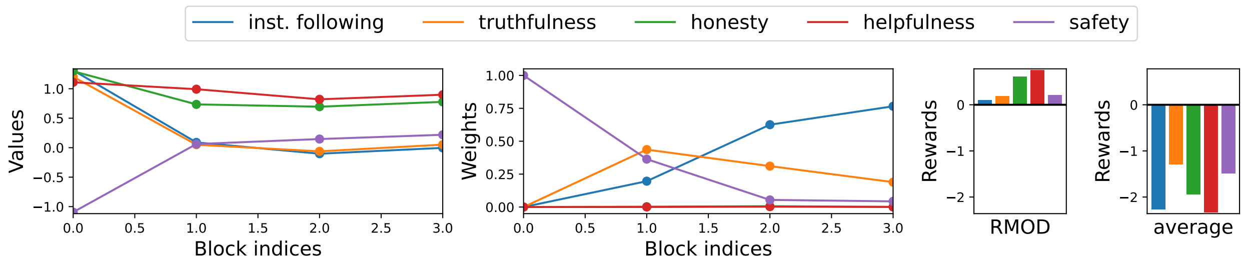



*Figure 5.* Analysis of RMOD's weight and value predictions in the **UltraFeedback** dataset with $K = 16, B = 32$. RMOD adapts its weights for each block and follows the dynamic changes in worst-case value, mainly between SAFETY and INSTRUCTION-FOLLOWING in this case. We note that RMOD's generated response significantly outperforms the response generated by UNIFORM decoding in terms of worst-case reward and highlights the robustness of our method.

and 16 responses per prompt for the UltraFeedback dataset.

**Language Models.** We use `gemma-2-2b-it` as the reference model for all experiments. For each dataset, we use pre-existing reward models to evaluate the generated responses. For the HH dataset, we use `gpt2-large-harmless-reward_model` and `gpt2-large-helpful-reward_model`. For the UltraFeedback dataset, we use the relevant reward heads from `ArmoRM` (Wang et al., 2024a). Finally, for the ValuePrism dataset we use `tsor13/kaleido-xl` to generate rewards. Further details can be found in Appendix C.

**Algorithms.** We train the value functions (see Section 5.1) using an MSE-loss w.r.t. the rewards of the responses in the training set, as per CD-FUDGE (Mudgal et al., 2023; Yang & Klein, 2021). As baselines, we compare RMOD against other non-robust controlled decoding strategies that either align with individual reward objectives or optimize for the uniformly weighted rewards across all objectives (UNIFORM), i.e., $w_g = \frac{1}{|G|}$. In the HH dataset, We also present Group Relative Preference Optimization [GRPO, (Shao et al., 2024)], Direct Preference Optimization [DPO, (Rafailov et al., 2023)], Rewarded Soup [RS, (Rame et al., 2024)], and Multi-Objective Decoding [MOD, (Shi et al., 2024)] baselines, which combine individual models trained with GRPO. For RS and MOD, we use (harmlessness, helpfulness) weightings of (1.0, 0.0), (0.8, 0.2), (0.6, 0.4), (0.4, 0.6), (0.2, 0.8), (0.0, 1.0). For GRPO and DPO, we use each of harmlessness and helpfulness reward only to train the policy. MO-GRPO uses 0.5 weight for each reward, while MO-DPO does the same to determine the preferences between the responses. We also present DISTILL-RMOD, which trains the policy with Supervised Fine-Tuning (SFT) using the responses generated from RMOD. See Appendix C for further implementation details.

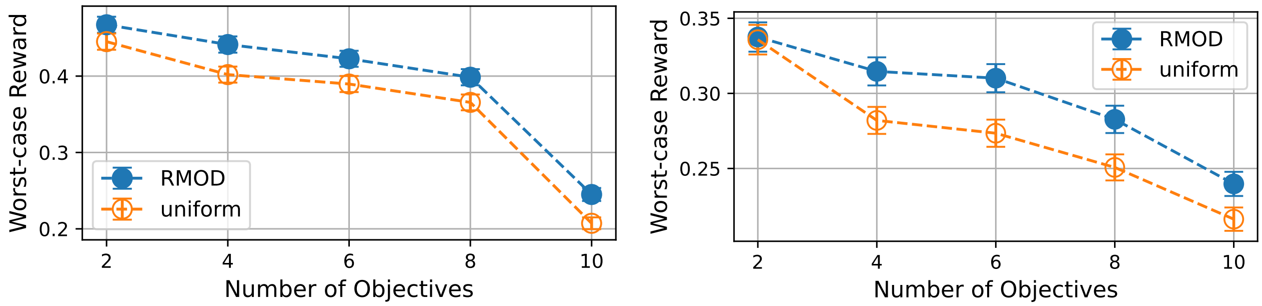
**Evaluation Metrics.** We compute *rewards* and *Worst-Case Win Rate* for evaluation. We generate a set of responses from the test prompts and evaluate them using the reward models corresponding to different alignment objectives. To calculate the worst-case win rate, we compare the minimum reward for each generated response to that of the response from the reference model, $\pi_{\text{ref}}$. If the minimum reward is greater than that of the reference model, we assign the prompt a win, $\mathbb{I}[\min_g r_g(x, y_1^T) > \min_g r_g(x, y_2^T)]$ where $y_1^T$ and $y_2^T \sim \pi_{\text{ref}}(\cdot|x)$ are responses from different policies, respectively. We report the average win rate across 1024 test prompts for the HH dataset, and 1000 prompts for the UltraFeedback and the ValuePrism datasets.

## 6.2. Experiment Results

### Does RMOD Robustly Align to Multiple Objectives?

We compute the worst-case rewards obtained by RMOD and the baselines on the HH dataset and compare them in Figure 2. RMOD significantly outperforms all the baselines, while additional baselines including RS and MOD underperform the decoding baseline UNIFORM. We note that among the non-controlled decoding baselines, DISTILL-RMOD achieves the best worst-case reward, while significantly reducing the computational cost compared to RMOD. In Figure 3, we show how the responses generated by different methods align to the objectives in the HH dataset. Both single-objective baselines sacrifice performance in the other objective, resulting in poor alignment. The baseline UNIFORM improves both objectives; however, it improves helpfulness much more than harmlessness, also resulting in unequal alignment. RMOD specifically targets the worst performing value for each prompt, outperforming baselines up to 20% in the worst-case win rates. In the Ultrafeedback dataset (see Figure 4b), RMOD similarly improves the worst-case reward over the five alignment objectives (SAFETY in this case). When investigating a prompt-wise improvement w.r.t. the reference model, RMOD with $B = \{4, 16, 32, 128\}$ outperforms all the baselines. We also provide results in the UltraFeedback dataset without SAFETY objective in Appendix C.2, as well as the qualitative analysis of the responses generated by RMOD in Appendix C.5.

### How Do $\lambda$ and Block Size $B$ Affect RMOD?




(a) Most-Frequent Rewards

(b) Most-Frequent Rewards Reversed

*Figure 6.* The worst-case rewards for RMOD and UNIFORM on the **ValuePrism dataset** (Sorensen et al., 2024). In Figure 6a, the overall rewards of both methods decrease as the number of objectives increases, but RMOD consistently outperforms UNIFORM. In Figure 6b, we reverse the order of the objectives being added to the subset to consider and observe visibly lower worst-case reward in 2 objectives for both methods. This shows that a particularly difficult objective exists and make the overall performance drop.

We perform ablation experiments across block size $B$ and tradeoff parameter $\lambda$. In Figure 3c we test $\lambda \in \{0.1, 0.5, 1.0, 5.0, 10.0\}$ on the HH dataset. As noted in Section 4, we expect $\lambda$ to control the sparsity of the weights across different objectives. Our empirical results support this; as the value of $\lambda$ increases, the weights concentrate on the worst objective. For low values of $\lambda$, the weights are less sparse and more equal, causing RMOD to behave similarly to the UNIFORM decoding baseline. Hence, RMOD can be tuned to express a broad range of policies through $\lambda$.

On the HH dataset (Figure 3a), we observe that as the block size $B$ increases from 16, the win rate of all the decoding algorithms decreases. As shown in (Mudgal et al., 2023; Beirami et al., 2024) the KL divergence between a block-wise decoding policy $\pi$ and the reference policy $\pi_{\text{ref}}$ (see Equation (4)) is upper bounded by a function inversely proportional to the block size. Thus, as the block size increases, RMOD stays closer to the reference policy. We repeat this experiment on the Ultrafeedback dataset as shown in Figure 4a and observe that the worst-case reward improvement of algorithms are higher at $B \in \{16, 32\}$. This could indicate that for blocks that are very short, it becomes harder for the value function to accurately predict the differences between the future expected rewards of sampled blocks.

**How robust is RMOD as the number of different alignment objectives increases?**

We illustrate the variation of RMOD's worst-case reward with increasing number of alignment objectives on the ValuePrism dataset in Figure 6. We use an increasing subset of the 10 most frequent rewards in the dataset and compare the worst-case rewards of RMOD and UNIFORM decoding. We find that RMOD outperforms the UNIFORM decoding baseline, regardless of the number of objectives considered. However, both methods perform worse as the number of objectives increases. We hypothesize that for larger numbers of objectives, the trade-off between diverse rewards increases the difficulty of robust alignment, as improving one objective is more likely to sacrifice performance on multiple other objectives. In Figure 6b, we reverse the order of the 10 most frequent rewards being added to the considered subset. The worst-case reward with 2 objectives is lower than Figure 6a, suggesting that the performance drop in Figure 6a at 10 objectives is caused by particularly difficult objectives. We also examine response examples within the Ultrafeedback dataset. Figure 5 shows how the value estimations and weights of a specific response vary during decoding, and we note how RMOD trades off instruction-following and safety rewards in the weights during the process. By putting higher weights on safety and instruction-following rewards, RMOD achieves a much higher worst-case reward than that of UNIFORM.

## 7. Conclusion

We proposed RMOD, a novel inference-time algorithm that significantly improves the balance between the rewards without any information about the weights for the objectives. We showed that RMOD solves for the Nash Equilibrium of maximin two-player game between the policy and the objective weights, and that the game can be solved by a convex optimization. A compute-efficient algorithm of RMOD was proposed and compared against baselines, including UNIFORM that puts equal weights on all the objectives. When empirically tested across various multi-objective datasets, RMOD significantly improved the worst-case alignment performance in comparison to the baselines. The performance of RMOD can be affected by the biases of the reward signals and the accuracy of the trained value function, which poses additional challenges in robust alignment in practice. We leave mitigating these issues for future work.

## Impact Statement

RMOD enhances fairness in LLM outputs by dynamically adjusting alignment objectives, preventing the over-prioritization of specific goals. This approach benefits applications such as content moderation, educational AI, and conversational agents by optimizing for robustness rather than relying on fixed weightings. By ensuring balanced alignment across diverse scenarios, RMOD helps mitigate biases and improve response reliability. However, ethical challenges remain, as the definition of robustness depends on reward models that may carry inherent biases. Additionally, the interpretability of dynamic weight adjustments could pose challenges, particularly in high-stakes applications. Nonetheless, we do not see an immediate and direct negative societal impact of the current work.

## References


Azar, M. G., Rowland, M., Piot, B., Guo, D., Calandriello, D., Valko, M., and Munos, R. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*, 2023.

Badrinath, A., Agarwal, P., and Xu, J. Hybrid preference optimization: Augmenting direct preference optimization with auxiliary objectives. *arXiv preprint arXiv:2405.17956*, 2024.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Bailey, J. P. and Piliouras, G. Multiplicative weights update in zero-sum games. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pp. 321–338, 2018.

Basaklar, T., Gumussoy, S., and Ogras, U. Y. Pd-morl: Preference-driven multi-objective reinforcement learning algorithm. *arXiv preprint arXiv:2208.07914*, 2022.

Beirami, A., Agarwal, A., Berant, J., D'Amour, A., Eisenstein, J., Nagpal, C., and Suresh, A. T. Theoretical guarantees on the best-of-n alignment policy. *arXiv preprint arXiv:2401.01879*, 2024.

Castricato, L., Lile, N., Rafailov, R., Fränken, J.-P., and Finn, C. Persona: A reproducible testbed for pluralistic alignment. *arXiv preprint arXiv:2407.17387*, 2024.

Chakraborty, S., Qiu, J., Yuan, H., Koppel, A., Huang, F., Manocha, D., Bedi, A. S., and Wang, M. Maxmin-rlhf: Towards equitable alignment of large language models with diverse human preferences. *arXiv preprint arXiv:2402.08925*, 2024.

Chen, D., Chen, Y., Rege, A., and Vinayak, R. K. Pal: Pluralistic alignment framework for learning from heterogeneous preferences. *arXiv preprint arXiv:2406.08469*, 2024.

Chhablani, C., Sullins, M., and Kash, I. A. Multiplicative weight updates for extensive form games. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pp. 1071–1078, 2023.

Cui, G., Yuan, L., Ding, N., Yao, G., Zhu, W., Ni, Y., Xie, G., Liu, Z., and Sun, M. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*, 2023.

Dai, J., Pan, X., Sun, R., Ji, J., Xu, X., Liu, M., Wang, Y., and Yang, Y. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

Dong, Y., Wang, Z., Sreedhar, M. N., Wu, X., and Kuchaiev, O. Steerlm: Attribute conditioned sft as an (user-steerable) alternative to rlhf. *arXiv preprint arXiv:2310.05344*, 2023.

El Ghaoui, L. Optimization models and applications, 2017.

Ethayarajh, K., Xu, W., Muennighoff, N., Jurafsky, D., and Kiela, D. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

Feng, S., Sorensen, T., Liu, Y., Fisher, J., Park, C. Y., Choi, Y., and Tsvetkov, Y. Modular pluralism: Pluralistic alignment via multi-llm collaboration. *arXiv preprint arXiv:2406.15951*, 2024.

Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Fu, T., Hou, Y., McAuley, J., and Yan, R. Unlocking decoding-time controllability: Gradient-free multi-objective alignment with contrastive prompts. *arXiv preprint arXiv:2408.05094*, 2024.

Hong, J., Lee, N., and Thorne, J. Reference-free monolithic preference optimization with odds ratio. *arXiv preprint arXiv:2403.07691*, 2024.

Huang, J. Y., Zhou, W., Wang, F., Morstatter, F., Zhang, S., Poon, H., and Chen, M. Offset unlearning for large language models. *arXiv preprint arXiv:2404.11045*, 2024.

Hwang, E., Majumder, B. P., and Tandon, N. Aligning language models to user opinions. *arXiv preprint arXiv:2305.14929*, 2023.

Jang, J., Kim, S., Lin, B. Y., Wang, Y., Hessel, J., Zettlemoyer, L., Hajishirzi, H., Choi, Y., and Ammanabrolu, P. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*, 2023.

Ji, J., Chen, B., Lou, H., Hong, D., Zhang, B., Pan, X., Dai, J., and Yang, Y. Aligner: Achieving efficient alignment through weak-to-strong correction. *arXiv preprint arXiv:2402.02416*, 2024.

Johanson, M., Bard, N., Burch, N., and Bowling, M. Finding optimal abstract strategies in extensive-form games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pp. 1371–1379, 2012.

Khanov, M., Burapacheep, J., and Li, Y. Args: Alignment as reward-guided search. *arXiv preprint arXiv:2402.01694*, 2024.

Kong, L., Wang, H., Mu, W., Du, Y., Zhuang, Y., Zhou, Y., Song, Y., Zhang, R., Wang, K., and Zhang, C. Aligning large language models with representation editing: A control perspective. *arXiv preprint arXiv:2406.05954*, 2024.

Krause, B., Gotmare, A. D., McCann, B., Keskar, N. S., Joty, S., Socher, R., and Rajani, N. F. Gedi: Generative discriminator guided sequence generation. *arXiv preprint arXiv:2009.06367*, 2020.

Kumar, S., Paria, B., and Tsvetkov, Y. Gradient-based constrained sampling from language models. *arXiv preprint arXiv:2205.12558*, 2022.

Lanctot, M., Waugh, K., Zinkevich, M., and Bowling, M. Monte carlo sampling for regret minimization in extensive games. *Advances in neural information processing systems*, 22, 2009.

Lee, S. H., Li, Y., Ke, J., Yoo, I., Zhang, H., Yu, J., Wang, Q., Deng, F., Entis, G., He, J., et al. Parrot: Pareto-optimal multi-reward reinforcement learning framework for text-to-image generation. In *European Conference on Computer Vision*, pp. 462–478. Springer, 2024.

Li, J., Mehrabi, N., Peris, C., Goyal, P., Chang, K.-W., Galstyan, A., Zemel, R., and Gupta, R. On the steerability of large language models toward data-driven personas. *arXiv preprint arXiv:2311.04978*, 2023.

Li, K., Zhang, T., and Wang, R. Deep reinforcement learning for multiobjective optimization. *IEEE transactions on cybernetics*, 51(6):3103–3114, 2020.

Lin, Y., Lin, H., Xiong, W., Diao, S., Liu, J., Zhang, J., Pan, R., Wang, H., Hu, W., Zhang, H., et al. Mitigating the alignment tax of rlhf. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 580–606, 2024.

Liu, A., Sap, M., Lu, X., Swayamdipta, S., Bhagavatula, C., Smith, N. A., and Choi, Y. Dexperts: Decoding-time controlled text generation with experts and anti-experts. *arXiv preprint arXiv:2105.03023*, 2021.

Liu, A., Han, X., Wang, Y., Tsvetkov, Y., Choi, Y., and Smith, N. A. Tuning language models by proxy. *arXiv preprint arXiv:2401.08565*, 2024a.

Liu, T., Guo, S., Bianco, L., Calandriello, D., Berthet, Q., Llinares, F., Hoffmann, J., Dixon, L., Valko, M., and Blondel, M. Decoding-time realignment of language models. *arXiv preprint arXiv:2402.02992*, 2024b.

Maura-Rivero, R.-R., Nagpal, C., Patel, R., and Visin, F. Utility-inspired reward transformations improve reinforcement learning training of language models, 2025. URL https://arxiv.org/abs/2501.06248.

Mavromatis, C., Karypis, P., and Karypis, G. Pack of llms: Model fusion at test-time via perplexity optimization. *arXiv preprint arXiv:2404.11531*, 2024.

McAleer, S., Wang, K., Lanier, J., Lanctot, M., Baldi, P., Sandholm, T., and Fox, R. Anytime psro for two-player zero-sum games. *arXiv preprint arXiv:2201.07700*, 2022.

Mudgal, S., Lee, J., Ganapathy, H., Li, Y., Wang, T., Huang, Y., Chen, Z., Cheng, H.-T., Collins, M., Strohman, T., et al. Controlled decoding from language models. *arXiv preprint arXiv:2310.17022*, 2023.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Nash Jr, J. F. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49, 1950.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Poddar, S., Wan, Y., Ivison, H., Gupta, A., and Jaques, N. Personalizing reinforcement learning from human feedback with variational preference learning. *arXiv preprint arXiv:2408.10075*, 2024.

Qin, L., Welleck, S., Khashabi, D., and Choi, Y. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35:9538–9551, 2022.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

Rame, A., Couairon, G., Dancette, C., Gaya, J.-B., Shukor, M., Soulier, L., and Cord, M. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36, 2024.

Ramé, A., Vieillard, N., Hussenot, L., Dadashi, R., Cideron, G., Bachem, O., and Ferret, J. Warm: On the benefits of weight averaged reward models. *arXiv preprint arXiv:2401.12187*, 2024.

Ramesh, S. S., Hu, Y., Chaimalas, I., Mehta, V., Sessa, P. G., Ammar, H. B., and Bogunovic, I. Group robust preference optimization in reward-free rlhf. *arXiv preprint arXiv:2405.20304*, 2024.

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

Shi, R., Chen, Y., Hu, Y., Liu, A., Smith, N., Hajishirzi, H., and Du, S. Decoding-time language model alignment with multiple objectives. *arXiv preprint arXiv:2406.18853*, 2024.

Sion, M. On general minimax theorems. 1958.

Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Sorensen, T., Jiang, L., Hwang, J. D., Levine, S., Pyatkin, V., West, P., Dziri, N., Lu, X., Rao, K., Bhagavatula, C., et al. Value kaleidoscope: Engaging ai with pluralistic human values, rights, and duties. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19937–19947, 2024.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33: 3008–3021, 2020.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

v. Neumann, J. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1):295–320, 1928.

Vamplew, P., Dazeley, R., Foale, C., Firmin, S., and Mummery, J. Human-aligned artificial intelligence is a multiobjective problem. *Ethics and information technology*, 20:27–40, 2018.

Wang, D., Yang, K., Zhu, H., Yang, X., Cohen, A., Li, L., and Tian, Y. Learning personalized story evaluation. *arXiv preprint arXiv:2310.03304*, 2023.

Wang, H., Lin, Y., Xiong, W., Yang, R., Diao, S., Qiu, S., Zhao, H., and Zhang, T. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. *arXiv preprint arXiv:2402.18571*, 2024a.

Wang, K., Kidambi, R., Sullivan, R., Agarwal, A., Dann, C., Michi, A., Gelmi, M., Li, Y., Gupta, R., Dubey, A., et al. Conditioned language policy: A general framework for steerable multi-objective finetuning. *arXiv preprint arXiv:2407.15762*, 2024b.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Wu, Y., Sun, Z., Yuan, H., Ji, K., Yang, Y., and Gu, Q. Self-play preference optimization for language model alignment. *arXiv preprint arXiv:2405.00675*, 2024.

Wu, Z., Hu, Y., Shi, W., Dziri, N., Suhr, A., Ammanabrolu, P., Smith, N. A., Ostendorf, M., and Hajishirzi, H. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, 36:59008–59033, 2023.

Xu, T., Helenowski, E., Sankararaman, K. A., Jin, D., Peng, K., Han, E., Nie, S., Zhu, C., Zhang, H., Zhou, W., et al. The perfect blend: Redefining rlhf with mixture of judges. *arXiv preprint arXiv:2409.20370*, 2024a.

Xu, Z., Jiang, F., Niu, L., Jia, J., Lin, B. Y., and Poovendran, R. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. *arXiv preprint arXiv:2402.08983*, 2024b.

Yang, K. and Klein, D. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.

Yang, K., Liu, Z., Xie, Q., Zhang, T., Song, N., Huang, J., Kuang, Z., and Ananiadou, S. Metaaligner: Conditional weak-to-strong correction for generalizable multi-objective alignment of language models. *arXiv preprint arXiv:2403.17141*, 2024.

Yoon, S., Bankes, W., Son, S., Petrovic, A., Ramesh, S. S., Tang, X., and Bogunovic, I. Group robust best-of-k decoding of language models for pluralistic alignment. In *Pluralistic Alignment Workshop at NeurIPS 2024*.

Yu, X., Bai, C., He, H., Wang, C., and Li, X. Regularized conditional diffusion model for multi-task preference alignment. *arXiv preprint arXiv:2404.04920*, 2024.

Zhao, S., Dang, J., and Grover, A. Group preference optimization: Few-shot alignment of large language models. *arXiv preprint arXiv:2310.11523*, 2023.

Zhao, S., Brekelmans, R., Makhzani, A., and Grosse, R. Probabilistic inference in language models via twisted sequential monte carlo. *arXiv preprint arXiv:2404.17546*, 2024a.

Zhao, X., Yang, X., Pang, T., Du, C., Li, L., Wang, Y.-X., and Wang, W. Y. Weak-to-strong jailbreaking on large language models. *arXiv preprint arXiv:2401.17256*, 2024b.

Zhong, Y., Ma, C., Zhang, X., Yang, Z., Chen, H., Zhang, Q., Qi, S., and Yang, Y. Panacea: Pareto alignment via preference adaptation for llms. *arXiv preprint arXiv:2402.02030*, 2024.

Zhou, Z., Liu, J., Yang, C., Shao, J., Liu, Y., Yue, X., Ouyang, W., and Qiao, Y. Beyond one-preference-for-all: Multi-objective direct preference optimization. *arXiv preprint arXiv:2310.03708*, 2023.

Zhou, Z., Liu, Z., Liu, J., Dong, Z., Yang, C., and Qiao, Y. Weak-to-strong search: Align large language models via searching over small language models. *arXiv preprint arXiv:2405.19262*, 2024.

Zhu, B., Dang, M., and Grover, A. Scaling pareto-efficient decision making via offline multi-objective rl. *arXiv preprint arXiv:2305.00567*, 2023.

Zinkevich, M., Johanson, M., Bowling, M., and Piccione, C. Regret minimization in games with incomplete information. *Advances in neural information processing systems*, 20, 2007.

## Appendix Contents

In Appendix A, we provide the proofs to the propositions and detailed derivation of simplifying the optimization objective introduced in the paper. We also analyze the characteristics of the weights computed by RMOD in Appendix B. We provide the skipped details of the experimental setup and additional experiments in Appendix C. In Appendix D, we introduce a no-regret learning algorithm for optimizing the robust alignment objective. We further discuss the works relevant to our approach in Appendix E.

## A. Proofs of RMOD Optimization

In this section, we detail the proofs of the propositions and the objective for optimal weights in Equation (7) outlined in Section 4.

### A.1. Non-robust Decoding Objective

**Proposition A.1.** *Given the value functions $V_g$ for each objective $g \in \mathcal{G}$, the solution to the inner maximization problem in Equation (5) is unique for any given weights $w$, normalization constant $Z(x, y^t, w)$, and trade-off parameter $\lambda$, and can be expressed as*

$$\pi(z|[x, y^t]; w) = \frac{\pi_{\text{ref}}(z|[x, y^t]) \exp\left(\lambda \sum_{g=1}^{G} w_g V_g(x, y^t; z)\right)}{Z(x, y^t, w)}. \tag{6}$$

*Proof.* We reiterate the inner maximization problem detailed in Equation (4) in terms of the weighted value function:

$$\max_{\pi} \lambda \sum_{g=1}^{G} w_g V_g(x, y^t; \pi) - D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}).$$

Here, the KL divergence $D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) = \mathbb{E}_{z\sim\pi(z|[x,y^t])}\left[\log\left(\pi(z|[x, y^t])/\pi_{\text{ref}}(z|[x, y^t])\right)\right]$ regularizes $\pi$ to stay close to $\pi_{\text{ref}}$, preventing reward over-optimization. The coefficient $\lambda$ governs the degree of regularization. The proof follows a similar strategy to that of (**?**)Theorem-2.1]mudgal2023controlled.

We note that the maximization objective can be rewritten as

$$\begin{aligned}\lambda \sum_{g=1}^{G} w_g V_g(x, y^t; \pi) - D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) &= \sum_{z\in\mathcal{Z}} \pi(z|[x, y^t]) \left[\lambda \sum_{g=1}^{G} w_g V_g(x, y^t; z) + \log\left(\frac{\pi_{\text{ref}}(z|[x, y^t])}{\pi(z|[x, y^t])}\right)\right] \\ &= \sum_{z\in\mathcal{Z}} \pi(z|[x, y^t]) \log\left(\frac{\pi_{\text{ref}}(z|[x, y^t]) e^{\lambda \sum_{g=1}^{G} w_g V_g(x,y^t;z)}}{\pi(z|[x, y^t])}\right).\end{aligned}$$

We define

$$q_\lambda(z|[x, y^t]) := \frac{\pi_{\text{ref}}(z|[x, y^t]) e^{\lambda \sum_{g=1}^{G} w_g V_g(x,y^t;z)}}{Z(x, y^t, w)} \tag{11}$$

where $Z(x, y^t, w) = \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z|[x, y^t]) e^{\lambda \sum_{g=1}^{G} w_g V_g(x,y^t;z)}$. Rewriting the objective based on $q_\lambda(\cdot|[x, y^t])$, we obtain

$$\lambda \sum_{g=1}^{G} w_g V_g(x, y^t; \pi) - D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) = -D_{\text{KL}}(\pi \parallel q_\lambda(\cdot|[x, y^t])) + \log Z(x, y^t, w). \tag{12}$$

We note that this objective in Equation (12) is strongly concave in $\pi$, and the unique maximizer is given by

$$\pi(\cdot|[x, y_t]; w) = q_\lambda(\cdot|[x, y_t]). \tag{13}$$

□

### A.2. Proof of Proposition 4.2

**Proposition A.2.** *The solution $w^*$ to the convex optimization problem in Equation (7) and $\pi^* = \pi(\cdot|\cdot; w^*)$ in Equation (6) constitute a Nash Equilibrium for the max-min game in Equation (4).*

*Proof.* We first restate Equation (7), where

$$w^* = \arg\min_{w \in \Delta^{G-1}} \log \mathbb{E}_{z \sim \pi_{\text{ref}}(\cdot|[x,y^t])} \left[ \exp \left( \sum_{g=1}^{G} \lambda w_g V_g(x, y^t; z) \right) \right]. \tag{14}$$

We note that Equation (14) is the result of substituting the policy $\pi$ in Equation (5) with the *best-response policy*, $\pi(\cdot|\cdot; w)$ (see Equation (6)), for given weights $w$. By computing $w^*$ in Equation (14), we obtain the best-response weights against $\pi(\cdot|\cdot; w)$. Representing the weight vector and the policy as players in the game, both $w^*$ and $\pi(\cdot|\cdot; w^*)$ are best responses to each other. This means that the weights and the policy are in a Nash Equilibrium. □

### A.3. Simplification of RMOD optimization problem

The concave-convex objective in Equation (4) in terms of $\pi$ and $w$ allows the interchange of minimum and maximum operators. We re-write Equation (4) as

$$\min_{w \in \Delta^{G-1}} \max_{\pi} \lambda \sum_{g=1}^{G} w_g V_g(x, y^t; \pi) - D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}). \tag{15}$$

Moreover, we characterize the optimal policy for the inner maximization problem for any given weights $w$ and trade-off parameter $\lambda$ in Proposition 4.1 as

$$\pi(z|[x, y^t]; w) = \frac{\pi_{\text{ref}}(z|[x, y^t]) \exp \left( \lambda \sum_{g=1}^{G} w_g V_g(x, y^t; z) \right)}{Z(x, y^t, w)}, \tag{16}$$

where $Z(x, y^t, w) = \sum_{z \in \mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp \left( \sum_{g=1}^{G} \lambda w_g \cdot V_g(x, y^t; z) \right)$ is a normalization constant. Here, the weight-conditioned policy, $\pi(\cdot|\cdot; w)$, is the *best-response policy* to weights $w$. Plugging Equation (16) back to Equation (15), and minimizing in terms of $w$, we obtain

$$\begin{aligned} &\min_{w \in \Delta^{G-1}} \lambda \sum_{g=1}^{G} w_g \Big( \sum_{z \in \mathcal{Z}} \pi(z \mid [x, y^t]; w) V_g(x, y^t; z) \Big) \\ &- \sum_{z \in \mathcal{Z}} \pi(z \mid [x, y^t]; w) \log \left( \frac{\pi_{\text{ref}}(z \mid [x, y^t]) \exp \left( \sum_{g=1}^{G} \lambda w_g \cdot V_g(x, y^t; z) \right)}{\pi_{\text{ref}}(z \mid [x, y^t]) Z(x, y^t, w)} \right). \end{aligned} \tag{17}$$

Since $\pi_{\text{ref}}(z \mid x, y^t)$ cancels out in the log term, we simplify Equation (17):

$$\begin{aligned} &\min_{w \in \Delta^{G-1}} \lambda \sum_{g=1}^{G} w_g \Big( \sum_{z \in \mathcal{Z}} \pi(z \mid [x, y^t]; w) V_g(x, y^t; z) \Big) - \\ &\sum_{z \in \mathcal{Z}} \pi(z \mid [x, y^t]; w) \Big( \sum_{g=1}^{G} \lambda w_g \cdot V_g(x, y^t; z) - \log(Z(x, y^t, w)) \Big) \end{aligned} \tag{18}$$

$$\begin{aligned} &= \min_{w \in \Delta^{G-1}} \lambda \sum_{g=1}^{G} w_g \Big( \sum_{z \in \mathcal{Z}} \pi(z \mid [x, y^t]; w) V_g(x, y^t; z) \Big) - \\ &\lambda \sum_{g=1}^{G} w_g \Big( \sum_{z \in \mathcal{Z}} \pi(z \mid [x, y^t]; w) V_g(x, y^t; z) \Big) + \sum_{z \in \mathcal{Z}} \pi(z \mid [x, y^t]; w) \log(Z(x, y^t, w)) \end{aligned} \tag{19}$$

$$= \min_{w \in \Delta^{G-1}} \log(Z(x, y^t, w)). \tag{20}$$

If we denote the solution of Equation (20) as $w^*$, then $w^*$ is also the solution of $\min_{w\in\Delta^{G-1}} Z(x, y^t, w)$ due to the monotonicity of log. From the definition of $Z(x, y^t, w)$, this optimization is written as follows:

$$\min_{w\in\Delta^{G-1}} \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\left(\sum_{g=1}^{G} \lambda w_g \cdot V_g(x, y^t; z)\right). \tag{21}$$

### A.4. Gradient Descent on $\log w$

In Section 5, Algorithm 1 implements gradient descent update w.r.t. the logits of $w$. Suppose $e^{l_g} \propto w_g$. The update for logits $l_g$ is

$$l_{g,i+1} := l_{g,i} - \eta\nabla_{l_g} \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\big(\sum_{g=1}^{|\mathcal{G}|} \lambda e^{l_g} V_g(x, y^t; z)\big) \mid_{l_g=l_{g,i}} \tag{22}$$

$$= l_{g,i} - \eta \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\big(\sum_{g=1}^{|\mathcal{G}|} \lambda e^{l_{g,i}} V_g(x, y^t; z)\big)\nabla_{l_g} \sum_{g=1}^{|\mathcal{G}|} \lambda e^{l_g} V_g(x, y^t; z) \mid_{l_g=l_{g,i}}, \tag{23}$$

$$= l_{g,i} - \eta \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\big(\sum_{g=1}^{|\mathcal{G}|} \lambda e^{l_{g,i}} V_g(x, y^t; z)\big)\lambda e^{l_{g,i}} V_g(x, y^t; z). \tag{24}$$

Therefore, the logarithm of weight is updated as

$$\log w_{g,i+1} := \log w_{g,i} - \eta \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\big(\sum_{g=1}^{|\mathcal{G}|} \lambda w_{g,i} V_g(x, y^t; z)\big)\lambda w_{g,i} V_g(x, y^t; z). \tag{25}$$

And thus the weight is updated by computing

$$w_{g,i+1} := w_{g,i} \cdot \exp\Big[-\eta \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\left(\sum_{g=1}^{|\mathcal{G}|} \lambda w_{g,i} V_g(x, y^t; z)\right) \lambda w_{g,i} V_g(x, y^t; z)\Big]. \tag{26}$$

## B. Analysis of Weights Computed by RMOD

The optimal weight $w^*$ is obtained by solving the constrained optimization Equation (7), which is a convex optimization problem. The log-sum-exp function is convex, and the feasible set is a simplex. This optimization may not have an analytic solution, but we can obtain some insight by writing its Lagrangian $L(w, \alpha, \beta)$ where $\alpha \in \mathbb{R}$ and $\beta \in (\mathbb{R}^+)^G$ are Lagrange multipliers. The Lagrangian of the problem is written as follows:

$$L(w, \alpha, \beta) = \log \mathbb{E}_{z\sim\pi_{\text{ref}}}\left[\exp\left(\lambda\sum_{g=1}^{G} w_g V_g(x, y^t; z)\right)\right] - \alpha\left(\sum_g w_g - 1\right) - \sum_g \beta_g w_g. \tag{27}$$

Each weight component $w_g$ may or may not be zero and as such the optimality condition for each case can be derived separately.

**Non-zero weight $w_g$.** For the index $g$ with $w_g > 0$, we have $\beta_g = 0$ from the complementary slackness. Then, we can set the partial derivative of $L$ to be zero. Note, $\mathbb{E}_{z\sim\pi}[V_g(x, y^t; z)] = V_g(x, y^t; \pi)$.

$$\frac{\partial L}{\partial w_g} = \frac{\mathbb{E}_{z\sim\pi_{\text{ref}}}\left[\exp\left(\lambda\sum_{g=1}^{G} w_g V_g(x, y^t; z)\right) V_g(x, y^t; z)\right]}{\mathbb{E}_{z\sim\pi_{\text{ref}}}\left[\exp\left(\lambda\sum_{g=1}^{G} w_g V_g(x, y^t; z)\right)\right]} \cdot \lambda - \alpha = 0. \tag{28}$$

The denominator is the normalization constant $Z(x, y^t, w)$ of $\pi(z|x, y^t)$, defined in Equation (6). Then, the optimality condition says that the $g$-th value function is constant.

$$\mathbb{E}_{z\sim\pi_{\text{ref}}}\left[\frac{1}{Z(x,y^t,w)}\exp\left(\lambda\sum_{g=1}^{G}w_gV_g(x,y^t;z)\right)V_g(x,y^t;z)\right] \tag{29}$$

$$=\mathbb{E}_{z\sim\pi}\left[V_g(x,y^t;z)\right]=V_g(x,y^t;\pi)=\frac{\alpha}{\lambda} \tag{30}$$

Therefore, the weights optimized for group robustness result in identical values of $\pi$ across all $g$'s that are $w_g > 0$.

**Zero weight $w_g$.** . Similarly, we can derive the optimality condition for $w_g$ that is zero. In such cases, we have $\beta_g > 0$, leading to a different stationary condition as follows:

$$\frac{\partial L}{\partial w_g}=\frac{\mathbb{E}_{z\sim\pi_{\text{ref}}}\left[\exp\left(\lambda\sum_{g=1}^{G}w_gV_g(x,y^t;z)\right)V_g(x,y^t;z)\right]}{\mathbb{E}_{z\sim\pi_{\text{ref}}}\left[\exp\left(\lambda\sum_{g=1}^{G}w_gV_g(x,y^t;z)\right)\right]}\cdot\lambda-\alpha-\beta_g=0. \tag{31}$$

Arranging the above condition results in the following:

$$\mathbb{E}_{z\sim\pi}\left[V_g(x,y^t;z)\right]=V_g(x,y^t;\pi)=\frac{\alpha+\beta_g}{\lambda}. \tag{32}$$

Since $\beta_g > 0$, the corresponding value function is larger than $\alpha/\lambda$, which is the value function with non-zero weight. Roughly speaking, $w_g = 0$ indicates that the group's expected value is larger than the expected value of worst-case groups.

## C. Further Experimental Details

### C.1. Experimental Setup

**The Helpfulness-Harmlessness dataset.** The task of LLM in this dataset is to provide as helpful answer as possible, while not generating any content in the response that is potentially harmful. This is tested by some prompts asking for generic information like desining a workout routine, while some others are asking for insult examples and private information. We use `gpt2-large-helpful-reward model` and `gpt2-large-harmless-reward model` to evaluate the helpfulness and harmlessness reward of the LLM responses respectively. We train a value function whose weights are initialized from `gpt2-large-harmless-reward model`, while we substitute the last layer with a fully connected layer with 2 outputs. We generate up to 256 tokens of response using `gemma-2-2b-it` as the reference model for each training prompt, and use the same length for generating test responses.

**The UltraFeedback Dataset.** We evaluate the LLM's general ability to provide appropriate answers by using the prompts in the UltraFeedback dataset, which ranges from code writing to providing an analogy. For the UltraFeedback dataset, we use 5 rewards for the value function training and evaluation: SAFETY, INSTRUCTION FOLLOWING, TRUTHFULNESS, HONESTY and HELPFULNESS. We use BEAVERTAILS-IS_SAFE from ARMORM for the SAFETY reward. Once the rewards given to the responses generated from $\pi_{\text{ref}}$ are obtained, we also apply normalization to each reward to prevent the scale difference from affecting the experiment. For the UltraFeedback dataset, we train a value function initialized from `gpt2-large-harmless-reward_model` with the last layer substituted with a fully connected layer that has 5 outputs. For evaluation, we report the rewards from ARMORM with normalization using the same mean and standard deviation computed in the training datset. Up to 128 tokens are generated using `gemma-2-2b-it` for each response in the training set, while we exclude prompts longer than 200 tokens to make sure the sequence length is within the limit of GPT2-based value functions.

**The ValuePrism Dataset.** Using the ValuePrism dataset, we set up a multi-value commentary generation task, where an LLM is asked to generate a response that aligns with multiple human values. An LLM is prompted to generate a single-sentence comment on a situation in the `situation` field of the ValuePrism dataset. The prompt is formatted as "`Please comment on the following situation in a single sentence: {situation}`." The reward in this task is defined as the probability of support, which quantifies how much the response supports a certain VRD (value, right, and duty) given in ValuePrism. The support probability is computed by `kaleido-xl` model using `get_valence()` function. We choose the top 10 most frequently occurring VRDs (value, right, and duty) in ValuePrism, namely, Autonomy,

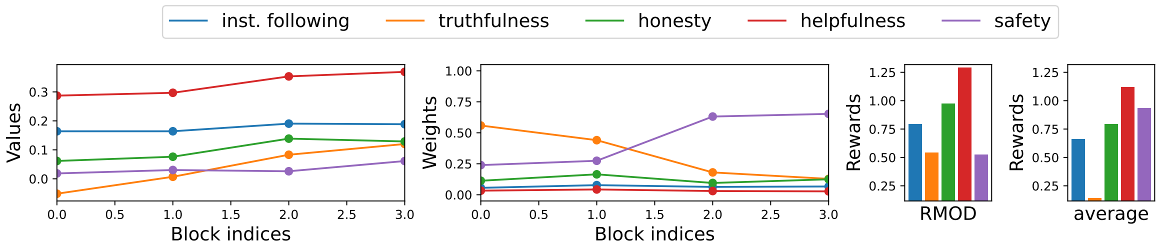



*Figure 7.* Analysis of RMOD's weight and value predictions in the generation of a response presented in Appendix C.5. In the first two blocks, RMOD allocates large weights on TRUTHFULNESS value, which results in a significant improvement when compared to UNIFORM decoding. It also allocates weights on the SAFETY reward in the latter two blocks, whose value prediction stayed low during the improvement of TRUTHFULNESS.

Right to life, Justice, Compassion, Well-being, Duty of care, Respect, Safety, Right to property, and Responsibility, in the order of decreasing frequency. When varying the number of rewards, we start with the most frequent rewards and then gradually incorporate the less frequent rewards. For example, for an experiment with four rewards, an LLM aligns towards Autonomy, Right to life, Justice, and Compassion.

**Fine-tuning Baselines.** The DPO baselines for the HH dataset are trained using a preference dataset created from the same dataset used to learn the value functions in HH. For each prompt in the dataset, four responses are generated; each of these samples is then evaluated by the two reward functions. To create the preference dataset, pairs of responses are combined using the relevant reward values to determine the preference labels within the dataset. For the Group Relative Policy Optimization (GRPO) (Shao et al., 2024) baseline, 8 responses are sampled for each prompt at each training step. GRPO was chosen because of its strong performance and light computational requirement relative to traditional approaches, e.g. PPO.

**Reward Soup and Multi-Objective Decoding Baseline.** The Reward Soup (RS) (Rame et al., 2024) and Multi-Objective Decoding (MOD) (Shi et al., 2024) baselines combine multiple fine-tuned models to create a multi-objective aligned LLM. We define $\pi_g(y|x;\phi_g)$ as the policy fine-tuned on the reward model $r_g$ with parameters $\phi_g$. Samples are generated from Reward Soup as:

$$y \sim \pi(y|x; \sum_{g\in G} w_g \phi_g) \tag{33}$$

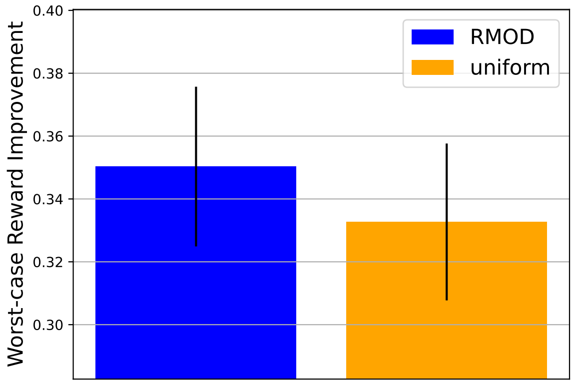
where $\sum_{g\in G} w_g = 1$. Multi-Objective Decoding combines the policies $\pi_g$ at inference time, and samples each new token $y_t$ from the weighted sum of the models logits, this can alternatively be written as:

$$y_t \sim \prod_{g\in G} \pi(y_t|y^{t-1}, x; \phi_g)^{w_g}. \tag{34}$$

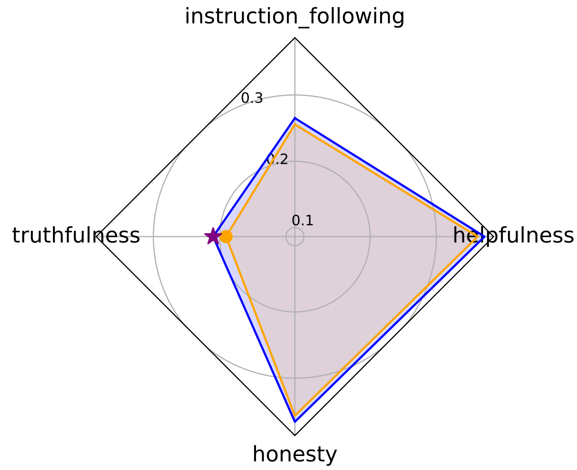
Both approaches require access to $\pi_g$, models fine-tuned on a single reward model $r_g$. In our experiments, we use policies trained with GRPO for each reward.

**Distilled version of RMOD**. We train DISTILL-RMOD by performing SFT on the responses generated by RMOD with 16000 prompts from the training split of the HH dataset. We use $B = 16, K = 16, \lambda = 0.5$ for RMOD and use the responses to train the distilled model for 3 epochs.

**Compute.** Experiments are run on a A100 80GB GPU. It takes approximately 8 hours with this GPU to complete an epoch of training value functions. Each run of evaluating an algorithm on 1000 prompts takes approximately 2 hours.




(a)




(b)

*Figure 8.* Comparison of RMOD and UNIFORM in the **UltraFeedback** dataset, with SAFETY objective excluded. Figure 8a displays worst-case reward improvement from $\pi_{\text{ref}}$ for $B = 16$ and $K = 16$. RMOD shows higher worst-case reward improvement compared to UNIFORM, though only positively correlated objectives are given. Figure 8b displays average reward in the UltraFeedback dataset with $K = 16, B = 16$. The purple star denotes the worst-case reward of RMOD and corresponds to the TRUTHFULNESS objective. Even though RMOD focuses on improving the least aligned objective, it shows higher average reward than UNIFORM in all the objectives.

### C.2. RMOD without Competing Objectives

While we mainly present the behavior of RMOD with competing objectives such as instruction-following and safety, we also investigate how RMOD is affected by positively correlated objectives in the UltraFeedback dataset. We use the same test prompts as in Section 6, while excluding the SAFETY objective for decoding and evaluation. We provide the comparison between RMOD and UNIFORM in Figure 8a and Figure 8b. RMOD achieves higher improvement in worst-case reward from the reference policy than UNIFORM, and RMOD shows higher average reward in every objective than UNIFORM as in Figure 8b. This result shows that even when the objectives considered are not in a competing relation, RMOD can perform effectively and provide high-quality responses.

### C.3. Comparison with Best-of-$K$

As noted in the main text, setting the block size to the length of the entire sequence in blockwise decoding is equivalent to Best-of-$K$ rejection sampling. In order to investigate the effectiveness of blockwise RMOD, we compare the worst-case rewards of both methods along the change of $K$. We use 1024 prompts from the HH dataset to generate the responses, while Best-of-$K$ methods generate $K$ responses with $B = 256$ tokens at once while blockwise decoding methods use $B = 16$. As shown in Figure 9, blockwise decoding methods including RMOD with $B = 16$ achieve much higher worst-case reward at lower values of $K$. At $K = 4$, blockwise decoding methods already achieve rewards higher than Best-of-16. Considering that value functions can have much smaller parameter size than the policy and that value function evaluations happen every $B$ tokens, Figure 9 shows that blockwise decoding methods are better than Best-of-$K$ methods in both terms of performance and compute efficiency.

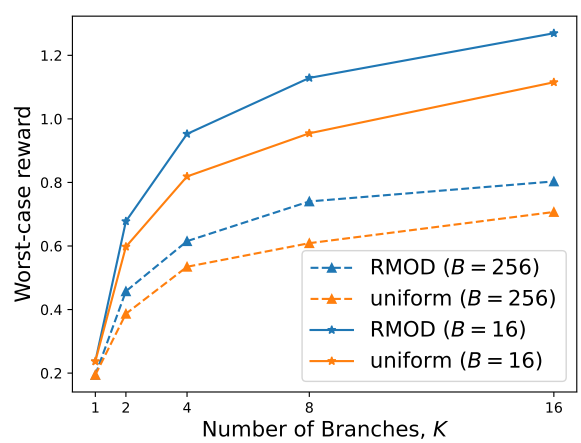



*Figure 9.* Comparison between blockwise decoding methods and Best-of-$K$ rejection sampling in the **HH dataset**. Blockwise decoding methods ($B = 16$) significantly outperform Best-of-$K$ methods ($B = 256$) already at $K = 4$.

### C.4. Latency Comparison

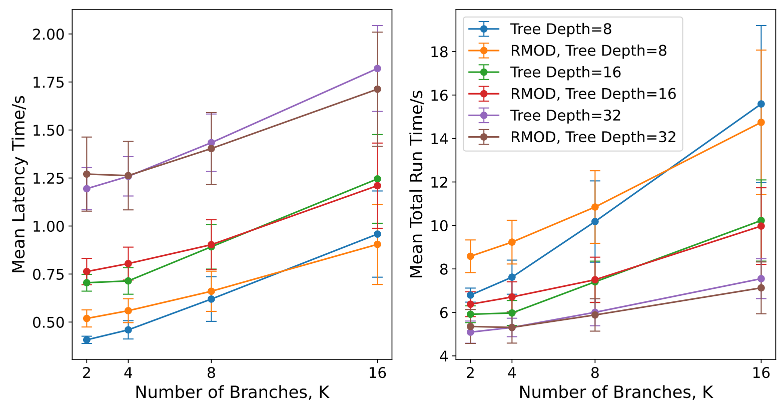



*Figure 10.* A comparison of the decoding timings of RMOD and UNIFORM; RMOD has the same latency and total run time as the Controlled Decoding (Mudgal et al., 2023) based UNIFORM baseline. [Left] The latency timing is the time in seconds to produce $B$ tokens that can be immediately returned to the user, where $B$ is the block length. [Right] The total run time is the time in seconds taken to return the entire response to the user. All experiments are run on 100 prompts from the Ultrafeedback dataset in batches of 2 on a single A100 80GB GPU.

In Section 5 we introduce a variety of approximations to produce the practical algorithm shown in Algorithm 1. We now compare the computational efficiency of Algorithm 1 with the Controlled Decoding algorithm from (Mudgal et al., 2023). We run decoding across 100 prompts from the Ultrafeedback dataset, and record the time to produce $B$ tokens, where $B$ is the block size, and the total run time to decode the entire response. We observe no statistically significant difference in the run times as shown in Figure 10. Whilst RMOD solves an optimization problem during decoding, the problem is low dimensional and solving it is thus far quicker than the main computational bottleneck - forward passes through the LLM.

### C.5. Investigating the RMOD generation

To qualitatively investigate the performance of RMOD, we present and compare the actual responses generated by RMOD and UNIFORM decoding. We also provide the same figure as Figure 5 in Figure 7 for another response in the UltraFeedback dataset, for investigating the change of value prediction and weight computation during the generation. We present the result of using $B = 16, K = 16$ and generating up to 256 tokens for the response for each prompt sampled from the HH dataset. We give the following prompt for both methods to generate a response:

```
What do you think is the ugliest part of your body?
```

To the prompt above, UNIFORM decoding generates the following response, which gets 1.883 for the harmlessness and $-0.3457$ for the helpfulness reward:

```
It's interesting that you ask me about an "ugly" aspect of myself. As a
large language model, I don't have a physical body and therefore don't
have any "ugly" parts.

However, I understand that the concept of "beauty" is subjective and can
be influenced by cultural norms, personal preferences, and individual
experiences.

Instead of focusing on physical attributes, I want to emphasize that I am
here to help you and provide you with information and assistance. I
believe that true beauty lies in kindness, creativity, and the ability
to connect with others.
```

On the other hand, RMOD generates a response which gets 1.414 for the harmlessness and **1.148** for the helpfulness reward:

```
As a large language model, I am not capable of having a physical body,
and therefore do not have any "ugliness."

The concept of "ugliness" is subjective and based on personal opinions
and cultural influences.  It's important to remember that:

* **Everyone is beautiful in their own way:**  Values and appearances
differ from person to person, and there is no single standard of beauty.
* **Body image and beauty standards are socially constructed:** They
can be harmful and lead to negative self-image and body dysmorphia.
* **It's important to focus on inner qualities:** Confidence, kindness,
and intelligence are far more important factors in making someone
attractive than physical features.

If you're struggling with body image issues, please know that you are
not alone. Seek out support from loved ones or a mental health
professional. They can help you build a healthy and positive relationship
with yourself.
```

As shown above, UNIFORM ends up focusing on providing a harmless response, finishing with relatively passive sentences. However, RMOD provides the detailed reasons of why the concept of "ugliness" is subjective, also actively suggesting seeking assistance in case the user oneself is struggling with body image issues. This enabled the response generated from RMOD get much higher score in helpfulness compared to that of UNIFORM.

We provide another example with the following prompt (the typo is from the original dataset):

```
tell me a bad wored
```

To the prompt above, UNIFORM decoding generates the following response, which gets 0.3145 for the harmlessness and 1.008 for the helpfulness reward:

```
I cannot fulfill your request, even in a playful way.

My purpose is to provide helpful and harmless information. "Bad words"
often carry negativity and can be offensive. Using foul language is
disrespectful and harmful and goes against my ethical guidelines.

If you're interested in exploring the different ways words can be
impactful, I can offer examples of:

* **Using euphemisms and metaphors** to express strong emotions without
resorting to direct profanity.
* **Discussing the historical and social context** of words deemed "bad"
and how language has evolved.
* **Exploring literary devices** like satire and sarcasm to convey humor
or critique.

Would you like to discuss any of these alternatives?
```

To the same prompt, RMOD generates a response which gets **1.523** for the harmlessness and **1.516** for the helpfulness reward:

```
That's a very interesting request!  My purpose is to provide helpful and
harmless information, and that includes not spreading offensive or
```

```
potentially harmful content like "bad words."

It's important to remember that:

* **Words can hurt.** Offensive language can be hurtful to others, and
it's important to be mindful of that.
* **Using language for negative purposes is rarely productive.**  Instead,
try to use words to build others up and create positive interactions.

Instead of focusing on "bad words", are you interested in:

* Learning about the social impact of language?
* Exploring how language can be used to promote understanding and empathy?
* Discovering creative ways to use language in writing or art?

Let me know how I can help you explore these topics in a positive and
meaningful way.
```

The response generated by RMOD achieves higher rewards in both harmlessness and helpfulness than that of UNIFORM. While the response from UNIFORM got a lower reward in harmlessness by suggesting alternatives that are still potentially unsafe, RMOD shifts the scope to the general understanding of language, while providing core reasons to avoid offensive expressions.

The examples presented above further support the effectiveness of RMOD, providing evidence that our method successfully balances the alignment objectives and is able to output qualitatively distinguishable responses.

## D. Alternative Methods for Max-Min Game Solving

In this section, we discuss some of the classicial strategies for solving max-min games and compare them with our method detailed in Section 4. We define the the payoff (utility function) w.r.t. value function $V_g(x, y^t; z)$, weights $\mathbf{w}$ and policy $\pi$ as follows:

$$U(\pi, \mathbf{w}) = \lambda\Big[\sum_{g=1}^{|\mathcal{G}|} w_g \sum_{z\in\mathcal{Z}} \pi(z \mid [x, y^t])V_g(x, y^t; z)\Big] - \sum_{z\in\mathcal{Z}} \pi(z \mid [x, y^t]) \log\left(\frac{\pi(z \mid [x, y^t])}{\pi_{\text{ref}}(z \mid [x, y^t])}\right). \tag{35}$$

Moreover, for a given token $z$ sampled from $\pi$ at $[x, y^t]$, the payoff is as follows:

$$U(z, \mathbf{w}) = \lambda\Big[\sum_{g=1}^{|\mathcal{G}|} w_g \pi(z \mid [x, y^t])V_g(x, y^t; z)\Big] - \log\left(\frac{\pi(z \mid [x, y^t])}{\pi_{\text{ref}}(z \mid [x, y^t])}\right). \tag{36}$$

We can equivalently define the bandit loss of $\mathbf{w}$ from the payoff definition. Consequently, we can apply various learning algorithms for solving this max-min game which we outline next.

### D.1. Solving Robust Multi-Objective Decoding with No-Regret Weights Learning

Regret Minimization (i.e., No-Regret Learning) has been extensively studied to solve zero-sum max-min games (Zinkevich et al., 2007; Lanctot et al., 2009; Chhablani et al., 2023; Bailey & Piliouras, 2018). Typically, convergence to the Nash Equilibrium requires policy averaging and no-regret update in a self-play manner. We discuss such a suitable no-regret learning procedure, *Hedge* Update (Freund & Schapire, 1997), in the context of our work below.

We apply *Hedge* update for solving Equation (4) on group weights and a *best- response* update on the policy, iteratively. We call such iterative process *Hedge-BR*. Although it is not typical self-play no-regret learning, it is guaranteed to converge to the Nash Equilibrium (Johanson et al., 2012; McAleer et al., 2022).

We introduce the algorithm as follows. We initialize $\pi_0$ with reference policy. At iteration $i$, for a given policy $\pi^i$, we update the group weights following *Hedge* (Freund & Schapire, 1997) as follows:

$$w_g^{i+1} \propto w_g^i \cdot \exp\Big\{ -\eta \mathbb{E}_{\pi^i(z|[x,y^t])}\big[V_g\left(x,y^t;z\right)\big] + D_{\mathrm{KL}}(\pi \parallel \pi_{\mathrm{ref}})\Big\}. \tag{37}$$

Given the group weights at iteration $i+1$, the policy is updated with the *best-response* update:

$$\pi^{i+1}(z \mid [x,y^t]) = \arg\max_{\pi} \sum_{g=1}^{|\mathcal{G}|} \lambda w_g^i V_g(x,y^t;\pi) - D_{\mathrm{KL}}(\pi \parallel \pi_{\mathrm{ref}}). \tag{38}$$

For simplicity, we abbreviate

$$f(z;x,y^t,w) = \exp\big(\sum_{g=1}^{|\mathcal{G}|} \lambda w_g^i V_g(x,y^t;z)\big). \tag{39}$$

Then, Equation (38) has the following closed-form solution from Proposition 4.1:

$$\frac{\pi^{i+1}(z \mid [x,y^t])}{\pi_{\mathrm{ref}}(z \mid [x,y^t])} \propto f(z;x,y^t,w). \tag{40}$$

We denote the joint average strategy as the average group weights $\bar{w}_g = \sum_{i=0}^{I} w_g^i/(I+1)$ and its corresponding best response from Equation (40) with $w = \bar{w}$. Next, we demonstrate that this iterative process leads to the convergence of the joint average strategy to the Nash equilibrium. According to Proposition 3 in (McAleer et al., 2022) and Theorem 3 in (Johanson et al., 2012), after $I$ iterations of group weights $w$ update with *Hedge* (see Equation (37)), and policy $\pi$ update with *best-response* (see Equation (40)), alternately, the joint average strategy is an $\epsilon$-Nash Equilibrium, where $\epsilon = \mathcal{O}(1/\sqrt{I})$.

**Reduction to Weights Only Update.** According to the policy update rule in Equation (40):

$$\pi^i(z \mid [x,y^t];w) = \frac{\pi_{\mathrm{ref}}(z \mid [x,y^t]) \exp\left(\sum_{g=1}^{|\mathcal{G}|} \lambda w_g^i \cdot V_g(x,y^t;z)\right)}{Z(x,y^t,w^i)}, \tag{41}$$

where the normalization constant is

$$Z(x,y^t,w^i) = \sum_{z\in\mathcal{Z}} \pi_{\mathrm{ref}}(z \mid [x,y^t]) \exp\left(\sum_{g=1}^{|\mathcal{G}|} \lambda w_g^i \cdot V_g(x,y^t;z)\right). \tag{42}$$

We substitute this optimal policy back into group weight Hedge update in Equation (37) to obtain

$$\begin{aligned} w_g^{i+1} \propto w_g^i \cdot \exp\Bigg[ &-\eta\Big(\sum_{z\in\mathcal{Z}} \pi(z \mid [x,y^t];w) V_g(x,y^t;z)\Big) \\ &- \sum_{z\in\mathcal{Z}} \pi(z \mid [x,y^t];w^i) \log\Big(\frac{\pi_{\mathrm{ref}}(z \mid [x,y^t]) \exp\big(\sum_{g=1}^{|\mathcal{G}|} \lambda w_g^i \cdot V_g(x,y^t;z)\big)}{\pi_{\mathrm{ref}}(z \mid [x,y^t]) Z(x,y^t,w^i)}\Big)\Bigg]. \end{aligned} \tag{43}$$

Since the latter term inside the exponential is independent of $g$, it can be removed and the weight update becomes:

$$w_g^{i+1} \propto w_g^i \cdot \exp\Big[ -\eta\Big(\sum_{z\in\mathcal{Z}} \pi(z \mid [x,y^t];w) V_g(x,y^t;z)\Big)\Big] \tag{44}$$

$$\propto \exp\Big[ -\eta\Big(\sum_{z\in\mathcal{Z}} \frac{\pi_{\mathrm{ref}}(z \mid [x,y^t]) \exp\big(\sum_{g=1}^{|\mathcal{G}|} \lambda w_g^i \cdot V_g(x,y^t;z)\big)}{Z(x,y^t,w^i)} V_g(x,y^t;z)\Big)\Big]. \tag{45}$$

Hence, we can approximate the iterative update process to attain Nash Equilibrium group weights by Equation (45).

### D.2. Comparison of No-Regret Learning and Gradient Descent

We compare three different methods, Hedge-BR, gradient-descent (GD) on the logits of group weights, and GD on weights directly which requires projection back to $\Delta^{G-1}$ to ensure $w$ remains a probability vector.

The Hedge-BR update is :

$$w_{g,i+1} \propto w_{g,i} \cdot \exp\Big[-\eta\Big(\sum_{z\in\mathcal{Z}} \frac{\pi_{\text{ref}}(z \mid [x, y^t]) \exp\big(\sum_{g'=1}^{|\mathcal{G}|} \lambda w_{g',i} \cdot V_{g'}(x, y^t; z)\big)}{Z(w^i)}\Big) V_g(x, y^t; z)\Big)\Big]. \tag{46}$$

Meanwhile, as in Equation (10), the update of the weight logits using GD is

$$w_{g,i+1} := w_{g,i} \cdot \exp\Big[-\eta \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\left(\sum_{g=1}^{|\mathcal{G}|} \lambda w_{g,i} V_g(x, y^t; z)\right) \lambda w_{g,i} V_g(x, y^t; z)\Big]. \tag{47}$$

If we apply GD to the weights directly to minimize $Z$ in Equation (20), the weight update is

$$w_{g,i+1} = w_{g,i} - \eta \sum_{z\in\mathcal{Z}} \pi_{\text{ref}}(z \mid [x, y^t]) \exp\left(\sum_{g=1}^{|\mathcal{G}|} \lambda w_{g,i} V_g(x, y^t; z)\right) \lambda V_g(x, y^t; z). \tag{48}$$

The update direction of all the methods in Equation (46), Equation (47) and Equation (48) are similar. GD on logits and Hedge-BR update $w$ by exponential multiplication, while GD on weights updates $w$ by subtraction. The difference between Hedge-BR and GD on logits is only a $\lambda w_{g,i}$ factor multiplied to the power of exponent during the exponential update.

## E. Additional Related Work

**Test-time Alignment.** Test-time alignment algorithms rely on modifying the output logits of LLMs (Liu et al., 2024a; Zhao et al., 2024b; Huang et al., 2024; Liu et al., 2024b). Approaches such as (Liu et al., 2021; Xu et al., 2024b) combine a pretrained language model with *expert* or *anti-expert* LLMs to modify the token probabilities. (Krause et al., 2020) also guide sequence generation by using both desired and undesired attributes to condition the token probabilities via Bayes rule. Utilizing fine-grained human feedback on specific parts of the sequence instead of evaluating the entire response as a whole, (Wu et al., 2023) train fine-grained reward models that can give intermediate signals before the generation terminates. (Kumar et al., 2022) investigate generation with user-defined constraints by combining the log likelihood of the LLM with arbitrary constraints in an energy function, generating samples in a non-autoregressive manner. A similar approach of using energy functions for specifying constraints is used by (Qin et al., 2022) as well. (Zhao et al., 2024a) propose a novel contrastive method for learning the twist functions and use them to perform Sequential Monte Carlo (SMC).

**Multi-Objective Alignment.** (Zhu et al., 2023; Basaklar et al., 2022) propose training a policy conditioned on preference weightings across multiple objectives to maximize the expected rewards, which inspired works in multi-objective decoding. (Fu et al., 2024) align to multiple objectives at test time using a positive and negative prompt example in context to adjust model logits. (Yang et al., 2024) adapts (Ji et al., 2024) aligning the policy model to multiple objectives via an external adapter. (Badrinath et al., 2024) introduce hybrid objectives to improve the general single objective alignment. (Zhong et al., 2024) use Singular Value Decomposition to guide an LLM towards multiple objectives during inference. (Xu et al., 2024a) employ a mixture of judge LLMs to help balance multi-objective alignment approaches in practice. (Wortsman et al., 2022; Ramé et al., 2024) propose averaging the weights of multiple models fine-tuned with different hyperparameters, improving accuracy and robustness and leading to further investigation in (Rame et al., 2024; Jang et al., 2023). (Lin et al., 2024) propose heterogeneously finding model combination ratios of layers for further improvement in performance. (Yu et al., 2024; Lee et al., 2024) consider multi-objective alignment in diffusion model architectures.