Inference-time alignment of language models by importance sampling on pre-logit space

Sekitoshi Kanai[1], **Tsukasa Yoshida**[1,2],**Kazumune Hashimoto**[3]
[1]NTT, Inc., [2]Toyohashi University of Technology, [3]The University of Osaka
sekitoshi.kanai@ntt.com

Abstract

Inference-time alignment of large language models (LLMs) attracts attentions because fine-tuning LLMs requires high computational costs. We propose a new sampling-based alignment method called adaptive importance sampling on pre-logits (AISP). AISP maximizes the expected value of a given reward model with respect to the distribution of pre-logits, which are outputs of the penultimate layer. We assume that the conditional distribution of pre-logits can be approximated by a Gaussian distribution, which bridges between the LLM alignment and the sampling-based control algorithm. The optimization of the pre-logit is solved by using importance sampling. AISP outperforms best-of-n sampling in terms of average rewards over the number of used samples.

1 Introduction

Alignment of large language models (LLMs) is a vital technique to enable the safe and widespread use of LLMs in real-world applications. A promising alignment method is reinforcement learning from human feedback (RLHF) [1–4]. However, RLHF imposes a heavy computational burden since fine-tuning LLMs requires huge computational costs [5–7]. To address the computational costs, inference-time (also known as test-time and decoding-time) alignment attracts attentions [6, 8–11].

Inference-time alignment aligns LLMs with human preference without updating parameters of LLMs. This paper focuses on inference-time alignment methods that find the optimal token sequences for the score of a given reward model. To this end, Mudgal et al. [12] proposed a method to control generation of tokens using prefix scorer module, which is a trained value function for the reward. Similarly, Kong et al. [6] formalized inference-time alignment as the control problem by editing representations with trained value functions. Though these approaches do not require training LLMs, they need to train value functions using a reward model: i.e., they still require the time-consuming training process including training dataset collection. As training-free inference-time alignment, best-of-n sampling (BoN) is a simple but effective method [9, 13–15]. BoN selects the response that achieves the highest reward values from N generated token sequences from the base LLMs. BoN asymptotically optimizes the same object function as KL-constrained reinforcement learning (RL) [16] under some condition. Though BoN achieves the optimal win-rate against KL divergence [17], there might be room for improvements in sample efficiency as it does not utilize non-optimal outputs during optimization.

In this paper, we propose adaptive importance sampling on pre-logits (AISP), which is a new sampling-based inference-time alignment using adaptive importance sampling. AISP optimizes the distribution of pre-logits, which are outputs of the penultimate layer, instead of the distribution of tokens. The pre-trained conditional distribution of the pre-logits given an output token should be an exponential family distribution because it naturally derives softmax function [18]. We further assume that the conditional distribution is a Gaussian distribution, which is used in the softmax-based image recognition [19]. Based on this idea, AISP injects Gaussian noise to the pre-logits and optimizes its

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Frontiers in Probabilistic Inference: Sampling Meets Learning.

mean. As a result, we formalize the alignment as the optimization problem of the parameters of a target Gaussian distribution with the KL constraint between the target distribution and the base LLMs. This formulation coincides with the sampling-based model predictive control with stochastic input signals called model predictive path integral control (MPPI) [20, 21]. Inspired by MPPI, we show that the lower bound of this problem is given by a certain free energy and solve this through adaptive importance sampling [22–24], which is an iterative importance sampling. Experiments demonstrate that AISP increases reward values faster than BoN in terms of the number of used generated samples.

2 Proposed method: AISP

To optimize the score of a reward model, we propose adaptive importance sampling on pre-logit space (AISP). AISP assumes that the pre-logit is obtained by the sum of the base LLM pre-logits and a Gaussian perturbation. The mean of perturbation is optimized through importance sampling to maximize reward values. This section first explains the problem of inference-time alignment. Next, we formalize our problem and explain the closed-solution, which is intractable distribution. To solve this problem, we present adaptive importance sampling for pre-logits.

Inference-time alignment for given reward models Let $x_t, y_t \in \mathcal{V}$ denote tokens in a vocabulary space $\mathcal{V}$ at the t-th position. Given an input prompt $\mathbf{x} = [x_1, \dots, x_{T_x}]$, an LLM generates a response $\mathbf{y} = [y_1, \dots, y_{T_y}]$ from the probability $P_{\mathrm{LLM}}(\cdot|\mathbf{x})$. We consider to generate aligned responses that maximize a given reward model $r(\mathbf{x}, \mathbf{y}) \in \mathbb{R}$. Inference-time alignment attempts to obtain aligned responses without model updates. BoN selects the best response $\mathbf{y}^*$ from the set of N generated responses $\mathcal{Y}_N$ as $y^* = \arg\max_{\mathbf{y} \in \mathcal{Y}_N} r(\mathbf{x}, \mathbf{y})$. Yang et al. [16] have shown that BoN asymptotically optimizes the following function used in RLHF:

$$\max_{\pi(\cdot|\mathbf{x})} \mathbb{E}_{\mathbf{y} \sim \pi(\cdot|\mathbf{x})} r(\mathbf{x}, \mathbf{y}) - \lambda D_{KL}(\pi(\cdot|\mathbf{x}) | P_{\mathrm{LLM}}(\cdot|\mathbf{x})). \tag{1}$$

2.1 Problem Formulation of AISP

To discuss the assumption in AISP, we first explain the output layer of neural language models. An auto-regressive LLM generally uses a softmax function with a linear layer as the last layer:

$$P_{\mathrm{LLM}}(y_t = y^i | \boldsymbol{y}_{<t}) = \frac{\exp(\boldsymbol{w}_i^\top \boldsymbol{z}_t + \boldsymbol{b}_i)}{\sum_{j=1}^{|\mathcal{V}|} \exp(\boldsymbol{w}_j^\top \boldsymbol{z}_t + \boldsymbol{b}_j)}, \tag{2}$$

where $\boldsymbol{z}_t \in \mathbb{R}^d$ is called a *pre-logit*, which is the output vector of the penultimate layer of the LLM: $\boldsymbol{z}_t = \phi_{\mathrm{LLM}}(\boldsymbol{y}_{<t})$ where $\boldsymbol{y}_{<t} = [y_1, \dots, y_{t-1}]$ are the past tokens.[1] Softmax function is naturally derived when the conditional distribution of pre-logits $p(\boldsymbol{z}_t | y_t = y^i)$ is an exponential family distribution.[2] From this fact, Lee et al. [19] assume that the pre-trained neural classifier has the pre-logtis following the Gaussian distribution given class label in image recognition. Since LLMs use softmax and cross-entropy loss similar to image recognition, we employ this assumption.

Once the t-th token y_t is given, the conditional distribution of pre-logit $p(\boldsymbol{z}_t | y_t)$ can be a Gaussian distribution. Inversely, we consider obtaining the optimal response through exploration of pre-logit distributions based on a Gaussian distribution. It enables us to optimize reward values with a tractable distribution. Since the pre-logit function $\phi_{\mathrm{LLM}}(\cdot)$ is a deterministic function, we inject a Gaussian noise $\boldsymbol{v}_i$ for the time interval $[0, \tau]$. Then, the response is written as

$$y_t = \arg\max_i \begin{cases} [\mathrm{softmax}(\boldsymbol{W}_{\mathrm{LLM}}(\phi_{\mathrm{LLM}}(\boldsymbol{y}_{<t}) + \boldsymbol{v}_t) + \boldsymbol{b}_{\mathrm{LLM}})]_i, \text{for } 0 < t \leq \tau, \\ [\mathrm{softmax}(\boldsymbol{W}_{\mathrm{LLM}} \phi_{\mathrm{LLM}}(\boldsymbol{y}_{<t}) + \boldsymbol{b}_{\mathrm{LLM}})]_i, \text{for } \tau < t, \end{cases} \tag{3}$$

where $\boldsymbol{W}_{\mathrm{LLM}}$ and $\boldsymbol{b}_{\mathrm{LLM}}$ are the parameter of the base LLM. The Gaussian noise $\boldsymbol{v}_t$ is sampled from $\mathcal{N}(\boldsymbol{u}_t, \sigma^2 \boldsymbol{I})$ where $\sigma^2 \in \mathbb{R}$ is a fixed constant and $\boldsymbol{u}_t$ is the parameter to be optimized. This can be regarded as that the pre-logit distribution is given by $p(\boldsymbol{z}_t | \boldsymbol{y}_{<t}) = \mathcal{N}(\phi_{\mathrm{LLM}}(\boldsymbol{y}_{<t}) + \boldsymbol{u}_t, \sigma^2 \boldsymbol{I})$ for $t \in [0, \tau]$. The distribution of a noise sequence $V = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_\tau] \in \mathbb{R}^{d \times \tau}$ is

$$q(V | U, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{d\tau}{2}}} \exp\left(-\frac{1}{2\sigma^2} \sum_{t=1}^{\tau} (\boldsymbol{v}_t - \boldsymbol{u}_t)^\top (\boldsymbol{v}_t - \boldsymbol{u}_t) \right), \tag{4}$$

[1] Generraly, the input prompt $\mathbf{x}$ corresponds to $y_1, \dots, y_{T_x}$.

[2] We consider a conditional distribution given an output token y_t not given the past tokens $\boldsymbol{y}_{<t}$.

where $U=[\boldsymbol{u}_1,\ldots,\boldsymbol{u}_T]\in\mathbb{R}^{d\times\tau}$. Let $\mathbb{Q}_{U,\sigma^2}$ be the distribution corresponding to $q(V|U,\sigma^2)$. Similar to Eq. (1), we optimize the expected reward values as

$$\min_U J(\mathbf{x},U)=\min_U -\mathbb{E}_{V\sim\mathbb{Q}_{U,\sigma^2}}\left[r(\mathbf{x},\mathbf{y}(V))\right]+\lambda\mathbb{D}_{\mathrm{KL}}(\mathbb{Q}_{U,\sigma^2}|\mathbb{P}) \tag{5}$$

where $\mathbf{y}(V)$ is a token sequence generated by Eq. (3) for $t=1,\ldots,T_y$. $\lambda\mathbb{D}_{\mathrm{KL}}(\mathbb{Q}|\mathbb{P})$ is the regularization term so that the resulting distribution does not deviate from the base LLM where $\lambda>0$ is a hyper-parameter. To this goal, a base distribution $\mathbb{P}$ should be zero-mean Gaussian distribution, which have the following density function: $p(V|\mathbf{0},\sigma^2)=1/(2\pi\sigma^2)^{\frac{d\tau}{2}}\exp\left(-1/2\sigma^2\sum_{t=1}^{\tau}\boldsymbol{z}_t^\top\boldsymbol{z}_t\right)$. After the optimization, we can obtain the optimal reward $\mathbf{y}_{\mathrm{AISP}}$ as $\mathbf{y}_{\mathrm{AISP}}=\mathbf{y}(U^*)$ where U^* is the solution of Eq. (5). Since we can generate several candidate responses at the last, we select $\mathbf{y}_{\mathrm{AISP}}=\mathrm{argmax}_{V\in\mathcal{V}}\,\mathbf{y}(V)$ where $\mathcal{V}=\{V^i|V^i\sim q(V|U^*,\sigma^2)\}$.

2.2 Free energy and Optimal distribution

Optimization problems such as Eq. (5) correspond to the optimal control problems with stochastic input signals $\boldsymbol{v}_t$ called model predictive path integral control (MPPI) [20, 21] and can be solved by considering a certain free energy. To optimize Eq. (5), we consider the following free energy:

$$F(r,p,\mathbf{x},\lambda)=\log\left(\mathbb{E}_{V\sim\mathbb{P}}\left[\exp\left(\tfrac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V))\right)\right]\right). \tag{6}$$

By using $\mathbb{Q}$ and Jensen's inequality, we have the following result:

Theorem 2.1. *Eq. (6) satisfies $-\lambda F(r,p,\mathbf{x},\lambda)\le J(\mathbf{x},U)$, and the equality holds if*

$$q^*(V)=\tfrac{1}{\eta}\exp\left(\tfrac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V))\right)p(V) \tag{7}$$

where η is a normalization constant as $\eta=\int_{\mathbb{R}^{d\times\tau}}\exp\left(\frac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V))\right)p(V)dV$.

The proof can be found in Appendix A. This theorem shows that the free energy Eq. (6) is the lower bound of the objective function Eq. (5), and the optimal distribution $\mathbb{Q}^*$ is given by Eq. (7). The optimal density function Eq. (7) is intractable, and it is difficult to be obtained directly. Next, we show how to approximate it by using importance sampling.

2.3 Adaptive importance sampling

We approximate $\mathbb{Q}^*$ of Eq. (7) by the Gaussian distribution $\mathbb{Q}_{U,\sigma^2}$ through importance sampling [25, 22–24]. To minimize $\mathbb{D}_{\mathrm{KL}}(\mathbb{Q}^*|\mathbb{Q}_{U,\sigma^2})$, the optimal mean $U^*=[\boldsymbol{u}_1^*,\ldots,\boldsymbol{u}_\tau^*]$ is obtained by

$$\boldsymbol{u}_t^*=\mathbb{E}_{\boldsymbol{z}_t\sim\mathbb{Q}^*}[\boldsymbol{v}_t]. \tag{8}$$

This computation is difficult because $\mathbb{Q}^*$ is an intractable distribution. To approximate this equation, we introduce a proposal density function $q(V|\hat{U},\sigma^2)$ and apply importance sampling as

$$\mathbb{E}_{V\sim\mathbb{Q}^*}[\boldsymbol{v}_t]=\int\boldsymbol{v}_t q^*(V)dV=\int\boldsymbol{v}_t\tfrac{q^*(V)}{q(V|\hat{U},\sigma^2)}q(V|\hat{U},\sigma^2)dV=\mathbb{E}_{V\sim\mathbb{Q}_{\hat{U},\sigma^2}}[w(V)\boldsymbol{v}_t], \tag{9}$$

where $\mathbb{Q}_{\hat{U},\sigma^2}$ is the distribution corresponding to $q(V|\hat{U},\sigma^2)$. This equation indicates that Eq. (8) can be approximated by $\boldsymbol{v}_t$ sampled from $\mathbb{Q}_{\hat{U},\sigma^2}$. The weight $w(V)=q^*(V)/q(V|\hat{U},\sigma^2)$ is computed by

$$w(V)=\tfrac{1}{\eta}\exp\left(\tfrac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V))-\tfrac{1}{\sigma^2}\sum_{t=1}^{\tau}\hat{\boldsymbol{u}}_t^\top\boldsymbol{v}_t-\tfrac{1}{2}\hat{\boldsymbol{u}}_t^\top\hat{\boldsymbol{u}}_t\right). \tag{10}$$

We generate n samples $\{V^i\}_{i=1}^n$ and approximate U^* as $\hat{\boldsymbol{u}}_t^*=\sum_i(w(V^i)/\sum_j w(V^j)\boldsymbol{v}_t^i)=\sum_i\bar{w}^i\boldsymbol{v}_t^i$ where $\sum_j w(V^j)$ is empirical normalization instead of η. The i-th weight $\bar{w}^i$ is given by

$$\bar{w}^i=\frac{\exp\left(\frac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V^i))-\frac{1}{\sigma^2}\sum_{t=1}^{\tau}\hat{\boldsymbol{u}}_t^\top\boldsymbol{v}_t^i\right)}{\sum_j\exp\left(\frac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V^j))-\frac{1}{\sigma^2}\sum_{t=1}^{\tau}\hat{\boldsymbol{u}}_t^\top\boldsymbol{v}_t^j\right)}, \tag{11}$$

which is implemented by using a softmax function. Note that the term of $\hat{\boldsymbol{u}}_t^\top\hat{\boldsymbol{u}}_t$ in Eq. (10) is canceled between the numerator and denominator. In Eq. (11), λ can be regarded as a temperature parameter. Small λ allows deviation from the base LLM, but large λ causes numerical instability [20]. To achieve both numerical stability and large penalties, we relax $\mathbb{P}$ as $p(V|\alpha\hat{U},\sigma)$ from $p(V|\mathbf{0},\sigma)$ where $0<\alpha<1$ by introducing the technique in MPPI [20]. Under this relaxation, the weight $\bar{w}^i$ becomes

$$\bar{w}^i=\frac{\exp\left(\frac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V^i))-\frac{1-\alpha}{\sigma^2}\sum_{t=1}^{\tau}\hat{\boldsymbol{u}}_t^\top\boldsymbol{v}_t^i\right)}{\sum_j\exp\left(\frac{1}{\lambda}r(\mathbf{x},\mathbf{y}(V^j))-\frac{1-\alpha}{\sigma^2}\sum_{t=1}^{\tau}\hat{\boldsymbol{u}}_t^\top\boldsymbol{v}_t^j\right)}. \tag{12}$$

Table 1: Average Rewards, diversity, and coherence of BoN and AISP.

Models	Methods	SHP			HH-RLHF		
		Reward	Diversity	Coherence	Reward	Diversity	Coherence
Llama3_8B & UltraRM	BoN (top-p)	-3.44 (0.07)	0.662 (0.007)	0.621 (0.005)	-5.24 (0.01)	0.730 (0.004)	0.608 (0.006)
	BoN ($\mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$)	-3.36 (0.06)	0.687 (0.01)	0.626 (0.006)	-5.22 (0.005)	**0.751** (0.005)	**0.614** (0.008)
	AISP	**-2.36** (0.07)	**0.705** (0.005)	**0.638** (0.005)	**-5.18** (0.01)	0.741 (0.002)	0.584 (0.002)
Vicuna_7B & UltraRM	BoN (top-p)	-3.97 (2)	0.860 (0.01)	0.656 (0.001)	-5.08 (0.1)	0.724 (0.1)	0.591 (0.05)
	BoN ($\mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$)	-2.17 (0.01)	0.870 (0.001)	0.656 (0.0004)	-5.00 (0.02)	**0.805** (0.005)	**0.632** (0.001)
	AISP	**-1.79** (0.003)	**0.872** (0.001)	**0.658** (0.001)	**-4.87** (0.01)	0.792 (0.004)	0.603 (0.0003)
Llama3_8B & Eurus	BoN (top-p)	-7.28 (0.03)	**0.750** (0.003)	0.641 (0.007)	-5.22 (0.004)	0.686 (0.006)	0.666 (0.006)
	BoN ($\mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$)	-7.22 (0.04)	0.714 (0.003)	0.645 (0.003)	-5.22 (0.01)	**0.714** (0.0004)	**0.676** (0.001)
	AISP	**-7.06** (0.04)	0.748 (0.004)	**0.656** (0.002)	**-5.19** (0.02)	**0.714** (0.001)	0.649 (0.002)
Vicuna_7B & Eurus	BoN (top-p)	-4.15 (0.03)	0.871 (0.002)	**0.656** (0.0001)	-5.03 (0.006)	0.769 (0.001)	0.637 (0.002)
	BoN ($\mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$)	-4.09 (0.02)	0.872 (0.001)	**0.656** (0.001)	-5.02 (0.005)	**0.800** (0.001)	**0.656** (0.001)
	AISP	**-3.95** (0.02)	**0.874** (0.002)	0.655 (0.0003)	**-4.95** (0.02)	0.792 (0.004)	0.625 (0.001)

We use these weights instead of Eq. (11) and obtain the optimal U^* can be obtained through importance sampling. However, importance sampling requires a lot of samples if the proposal distribution $q(V|\hat{U}, \sigma^2)$ is far from $\mathbb{Q}^*$. To mitigate this issue, we employ the adaptive importance sampling [22–24], which iteratively updates $\hat{\boldsymbol{u}}_t$ by using importance sampling. We provide a pseudo code of AISP and explain the detailed implementation in Appendix B,.

3 Experiments

3.1 Setup

To evaluate the performance of AISP, we use Anthropic's HH-RLHF [4] and Stanford human preferences (SHP) datasets [26], which are used to align LLMs for helpfulness and harmlessness, following [6]. We use randomly selected 1000 entries of the test datasets due to limited computation resources like [27]. We use Vicuna-7B-v1.5 [28][3] and Llama-3-8B [29] as the base LLMs, and reward models are UltraRM-13b (UltraRM) [30] and Eurus-RM-7b (Eurus) [31]. Hyper-parameters of AISP is shown in Appendix C.1 n, τ, and $\kappa = 32$ are set to 32, 128, and 32, respectively. A baseline method is BoN with $N = \kappa n = 1024$ samples. We use two generating strategies to construct candidate sets $\mathcal{Y}_N$ in BoN: *BoN (top-p)* uses top-p (nucleus) sampling with temperature of 0.6 and $p = 0.9$, which are default parameters of Llama3. *BoN ($\mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$)* injects Gaussian noises following $\mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$ into pre-logits like AISP and applies greedy search for each injected pre-logits to construct $\mathcal{Y}_N$. The evaluation metrics are the reward values by UltraRM, coherence, diversity.

3.2 Results

Table 1 lists average rewards, diversity score, and coherence score of each method. Values are presented as mean (standard deviation) for three trials. In terms of average reward, AISP achieves the highest among methods. AISP achieved up to about 30% improvement over BoN (top-p). Thus, AISP is superior to BoN as a sample-based reward optimization. Additionally, BoN with gaussian pre-logit noise (BoN ($\mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$)) achieves higher rewards than BoN (top). Thus, the Gaussian distribution of pre-logits can generate better candidate sets $\mathcal{Y}_N$. This might imply that the Gaussian assumption is not very strong, and it generates good candidate responses even without importance sampling. In terms of diversity and coherence scores, AISP does not always outperform BoN. This might be because reward models do not prioritize these perspectives. Even so, these scores are also related to the quality of responses, and it is difficult to conclude that AISP is superior to BoN based solely on the average rewards. Thus, we will evaluate responses by using GPT-4 in Appendix C.

4 Conclusion

In this paper, we propose adaptive importance sampling on a pre-logit distribution for alignment of LLMs. AISP assumes that pre-logit distributions are composed of a Gaussian perturbation and the pre-logit of the base LLM, and optimizes the Gaussian perturbation through importance sampling.

[3] `https://huggingface.co/lmsys/vicuna-7b-v1.5`

References

[1] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[2] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Proc. NeurIPS*, 30, 2017.

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

[4] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[5] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[6] Lingkai Kong, Haorui Wang, Wenhao Mu, Yuanqi Du, Yuchen Zhuang, Yifei Zhou, Yue Song, Rongzhi Zhang, Kai Wang, and Chao Zhang. Aligning large language models with representation editing: A control perspective. *Proc. NeurIPS*, 37:37356–37384, 2024.

[7] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1 (2):3, 2022.

[8] Bolian Li, Yifan Wang, Anamika Lochab, Ananth Grama, and Ruqi Zhang. Cascade reward sampling for efficient decoding-time alignment. *arXiv preprint arXiv:2406.16306*, 2024.

[9] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

[10] Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Dylan J Foster, and Akshay Krishnamurthy. Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment. *arXiv preprint arXiv:2503.21878*, 2025.

[11] Yuhui Li, Fangyun Wei, Jinjing Zhao, Chao Zhang, and Hongyang Zhang. RAIN: Your language models can align themselves without finetuning. In *Proc. ICLR*, 2024.

[12] Sidharth Mudgal, Jong Lee, Harish Ganapathy, Yaguang Li, Tao Wang, Yanping Huang, Zhifeng Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, et al. Controlled decoding from language models. In *Proc. ICML*, pages 36486–36503. PMLR, 2024.

[13] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

[14] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

[15] Pier Giuseppe Sessa, Robert Dadashi-Tazehozi, Leonard Hussenot, Johan Ferret, Nino Vieillard, Alexandre Rame, Bobak Shahriari, Sarah Perrin, Abram L. Friesen, Geoffrey Cideron, Sertan Girgin, Piotr Stanczyk, Andrea Michi, Danila Sinopalnikov, Sabela Ramos Garea, Amélie Héliou, Aliaksei Severyn, Matthew Hoffman, Nikola Momchev, and Olivier Bachem. BOND: Aligning LLMs with best-of-n distillation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[16] Joy Qiping Yang, Salman Salamatian, Ziteng Sun, Ananda Theertha Suresh, and Ahmad Beirami. Asymptotics of language model alignment. In *2024 IEEE International Symposium on Information Theory (ISIT)*, pages 2027–2032. IEEE, 2024.

[17] Lin Gui, Cristina Garbacea, and Victor Veitch. BoNBon alignment for large language models and the sweetness of best-of-n sampling. In *Proc. NeurIPS*, 2024.

[18] Christopher M Bishop. *Neural Networks for Pattern Recognition*. Oxford university press, 1995.

[19] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.

[20] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Information-theoretic model predictive control: Theory and applications to autonomous driving. *IEEE Transactions on Robotics*, 34(6):1603–1622, 2018.

[21] Grady Williams, Andrew Aldrich, and Evangelos A Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40 (2):344–357, 2017.

[22] Teun Kloek and Herman K Van Dijk. Bayesian estimates of equation system parameters: an application of integration by monte carlo. *Econometrica: Journal of the Econometric Society*, pages 1–19, 1978.

[23] Olivier Cappé, Arnaud Guillin, Jean-Michel Marin, and Christian P Robert. Population monte carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929, 2004.

[24] Monica F Bugallo, Victor Elvira, Luca Martino, David Luengo, Joaquin Miguez, and Petar M Djuric. Adaptive importance sampling: The past, the present, and the future. *IEEE Signal Processing Magazine*, 34(4):60–79, 2017.

[25] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.

[26] Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. Understanding dataset difficulty with $\mathcal{V}$-usable information. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008. PMLR, 17–23 Jul 2022.

[27] Yuu Jinnai, Tetsuro Morimura, Kaito Ariu, and Kenshi Abe. Regularized best-of-n sampling to mitigate reward hacking for language model alignment. In *ICML 2024 Workshop on Models of Human Feedback for AI Alignment*, 2024.

[28] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

[29] AI@Meta. Llama 3 model card. 2024. URL `https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md`.

[30] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback, 2023.

[31] Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, Zhenghao Liu, Bowen Zhou, Hao Peng, Zhiyuan Liu, and Maosong Sun. Advancing llm reasoning generalists with preference trees, 2024.

A Proofs

A.1 Proof of Theorem 2.1

Theorem. Free energy Eq. (6) satisfies the following inequality

$$-\lambda F(r, p, \mathbf{x}, \lambda) \leq J(\mathbf{x}, U), \tag{13}$$

and the equality holds if

$$q^*(V) = \frac{1}{\eta} \exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) p(V) \tag{14}$$

where η is a normalization constant as

$$\eta = \int_{\mathbb{R}^{d\times\tau}} \exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) p(V) dV. \tag{15}$$

Proof. Similar to importance sampling, F can be written by using $\mathbb{Q}$ as

$$F(r, p, \mathbf{x}, \lambda) = \log\left(\int \exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) \frac{q(V)}{q(V)} p(V) dV\right) \tag{16}$$

$$= \log\left(\int \exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) \frac{p(V)}{q(V)} q(V) dV\right) \tag{17}$$

$$= \log\left(\mathbb{E}_{V\sim\mathbb{Q}}\left[\exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) \frac{p(V)}{q(V)}\right]\right) \tag{18}$$

From Jensen's inequality, we have

$$F(r, p, \mathbf{x}, \lambda) = \log\left(\mathbb{E}_{V\sim\mathbb{Q}}\left[\exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) \frac{p(V)}{q(V)}\right]\right) \tag{19}$$

$$\geq \mathbb{E}_{V\sim\mathbb{Q}}\left[\log\left(\exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) \frac{p(V)}{q(V)}\right)\right] \tag{20}$$

$$= \mathbb{E}_{V\sim\mathbb{Q}}\left[\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V)) - \log\left(\frac{q(V)}{p(V)}\right)\right] \tag{21}$$

$$= \mathbb{E}_{V\sim\mathbb{Q}}\left[\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right] - D_{\mathrm{KL}}(\mathbb{Q}|\mathbb{P}) \tag{22}$$

$$\tag{23}$$

Multiplying both sides of each equation by $-\lambda$, we have the following:

$$-\lambda F(r, p, \mathbf{x}, \lambda) \leq -\mathbb{E}_{V\sim\mathbb{Q}}\left[r(\mathbf{x}, \mathbf{y}(V))\right] + \lambda D_{\mathrm{KL}}(\mathbb{Q}|\mathbb{P}). \tag{24}$$

Next, we substituting Eq. (7) into KL divergence as:

$$D_{\mathrm{KL}}(\mathbb{Q}|\mathbb{P}) = \int \log\left(\frac{q(V)}{p(V)}\right) q^*(V) dV \tag{25}$$

$$= \int \log\left(\frac{\frac{1}{\eta}\exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) p(V)}{p(V)}\right) q^*(V) dV \tag{26}$$

$$= \int \log\frac{1}{\eta}\exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) q^*(V) dV \tag{27}$$

$$= -\log(\eta) + \int \frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V)) q^*(V) dV \tag{28}$$

$$= -\log(\eta) + \frac{1}{\lambda}\mathbb{E}_{V\sim\mathbb{Q}^*}\left[r(\mathbf{x}, \mathbf{y}(V))\right]. \tag{29}$$

$-\log(\eta)$ becomes $-F(r, p, \mathbf{x}, \lambda)$ as

$$-\log(\eta) = -\log\left(\int_{\mathbb{R}^{d\times\tau}} \exp\left(\frac{1}{\lambda} r(\mathbf{x}, \mathbf{y}(V))\right) p(V) dV\right) \tag{30}$$

$$= -F(r, p, \mathbf{x}, \lambda) \tag{31}$$

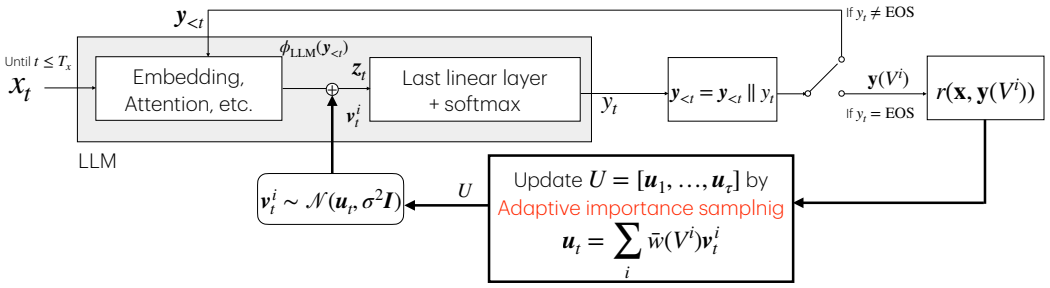


Figure 1: Illustration of AISP.

Algorithm 1 Pseudo code of AISP

Require: Hyper-parameters λ, α, σ^2, n, and κ. reward models $r(\mathbf{x}, \mathbf{y})$, Input prompt $\mathbf{x}$

1: Initialization: $\hat{U} = \boldsymbol{O}$, $r_{\mathrm{best}} = -\infty$
2: **for** $k = 1, \ldots, \kappa$ **do**
3: **for** $i = 1, \ldots, n$ **do**
4: $V^i \sim q(V|\hat{U}, \sigma^2)$
5: $\boldsymbol{y}_{<1}^i = \mathbf{x}$ for $i = 1, \ldots, n$
6: **for** $t = 1, \ldots T$ **do**
7: We get $\boldsymbol{z}_t = \phi_{\mathrm{LLM}}(\boldsymbol{y}_{<t})$ by adding $\boldsymbol{y}_{<t}^i$ to LLM
8: **if** $t \le \tau$ **then**
9: $\boldsymbol{z}_t = \phi_{\mathrm{LLM}}(\boldsymbol{y}_{<t}) + \boldsymbol{v}_t^i$
10: $y_t^i = \mathrm{argmax}_j[\mathrm{softmax}(\boldsymbol{W}_{\mathrm{LLM}} \boldsymbol{z}_t + \boldsymbol{b}_{\mathrm{LLM}})]_j$
11: $\boldsymbol{y}_{<t+1}^i = \boldsymbol{y}_{<t}^i \| y_t^i$
12: **if** $y_t^i = \mathrm{EOS}$ **then**
13: $\mathbf{y}(V^i) = \boldsymbol{y}_{<t+1}^i$ and break
14: Get rewards $r(\mathbf{x}, \mathbf{y}(V^i))$ by adding $\mathbf{y}(V^i)$ to the reward model
15: **if** $r_{\mathrm{best}} < r(\mathbf{x}, \mathbf{y}(V^i))$ **then**
16: $\mathbf{y}_{\mathrm{best}} = \mathbf{y}(V^i)$ and $r_{\mathrm{best}} = r(\mathbf{x}, \mathbf{y}(V^i))$
17: Compute weights $\bar{w}^i$ by Eq. (12) for $i = 1, \ldots, n$
18: Update $\hat{U} = [\hat{\boldsymbol{u}}_1, \ldots, \hat{\boldsymbol{u}}_\tau]$ by $\hat{\boldsymbol{u}}_t = \sum_i \bar{w}^i \boldsymbol{v}_t^i$
19: $U^* = \hat{U}$ and generate $\mathbf{y}(U^*) = \mathrm{argmax}_{V \in \mathcal{V}} \mathbf{y}(V)$ where $\mathcal{V} = \{V^i | V^i \sim q(V|U^*, \sigma^2)\}$
20: **if** $r_{\mathrm{best}} < r(\mathbf{x}, \mathbf{y}(U^*))$ **then**
21: $\mathbf{y}_{\mathrm{best}} = \mathbf{y}(U^*)$ and $r_{\mathrm{best}} = r(\mathbf{x}, \mathbf{y}(U^*))$
22: **Return** $\mathbf{y}_{\mathrm{best}}$ and r_{best}

Therefore, Eq. (29) becomes

$$D_{\mathrm{KL}}(\mathbb{Q}|\mathbb{P}) = -F + \frac{1}{\lambda} \mathbb{E}_{V \sim \mathbb{Q}^*} \left[r(\mathbf{x}, \mathbf{y}(V)) \right]. \tag{32}$$

and thus, we have

$$-\lambda F = -\mathbb{E}_{V \sim \mathbb{Q}^*} \left[r(\mathbf{x}, \mathbf{y}(V)) \right] + \lambda D_{\mathrm{KL}}(\mathbb{Q}|\mathbb{P}). \tag{33}$$

which completes the proof. □

B Implementation of AISP

Fig. 1 shows the outline of AISP. In this section, we will explain the detailed algorithm and implementation of AISP.

Algorithm 1 is the pseudo code of AISP. First, we generate V^i from the prior distribution in Line 5 and generate responses $y(V^i)$ in Lines 6-13. Line 10 decodes a token based on pre-logit. Since we observed that statistical sampling degrades the performance of AISP, we use a deterministic greedy

Table 2: Win rate for AISP vs BoN (top-p).

Datasets	Models	Rate AISP	Draw	BoN
SHP	Llama3_8B & UltraRM	**51.7**	7.67	40.7
	Vicuna_7B& UltraRM	**43.0**	35.3	21.7
	Llama3_8B &Eurus	**51.7**	6.67	41.7
	Vicuna_7B& Eurus	**26.7**	48.3	25.3
HH-RLHF	Llama3_8B& UltraRM	**44.0**	14.0	42.0
	Vicuna_7B& UltraRM	**55.3**	20.0	24.7
	Llama3_8B& Eurus	**47.0**	8.33	44.7
	Vicuna_7B& Eurus	**44.7**	25.0	30.0

Table 3: Win rate for AISP vs BoN ($\mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$).

Datasets	Models	Rate AISP	Draw	BoN
SHP	Llama3_8B & UltraRM	**49.0**	7.33	43.7
	Vicuna_7B& UltraRM	**36.3**	38.0	25.7
	Llama3_8B &Eurus	**45.7**	9.67	44.7
	Vicuna_7B& Eurus	**30.7**	45.3	24.0
HH-RLHF	Llama3_8B& UltraRM	43.3	11.7	**45.0**
	Vicuna_7B& UltraRM	**37.0**	28.0	35.0
	Llama3_8B& Eurus	**46.0**	8.0	**46.0**
	Vicuna_7B& Eurus	**41.7**	20.3	38.0

search. Next, we evaluate reward values for each $\mathbf{y}(V^i)$ in Line 14. Line 15 stores the best response during AISP because we select the best $\mathbf{y}$ among $n\kappa$ samples as the results like BoN. After reward evaluation, we update $\hat{U}$ in Lien 18. Finally, we generate $\mathbf{y}(U^*)$ as a last candidate of response and compare it with $\mathbf{y}_{\mathrm{best}}$. Note that though adaptive importance sampling generally uses $n\kappa$ samples, i.e., all generating samples, at the last iteration, we only use the n generated samples for each iteration due to computational cost. If we need multiple responses, AISP can be modified to output the set of $\mathbf{y}$ by using top-k in Lines 16 and 21.

Parallelization Adaptive importance sampling contains both parallel and sequential processes. We generate $\mathbf{y}(V^i)$ for $i = 1, \ldots, n$ in parallel, this can be implemented as a mini-batch computation for a prompt $\mathbf{x}$. Iterations $k = 1, \ldots, \kappa$ should be executed sequentially. Therefore, n and κ increase space-complexity and time-complexity, respectively, and they should be tuned according to practical needs and performance.

C Detailed experimental setup and additional experimental results

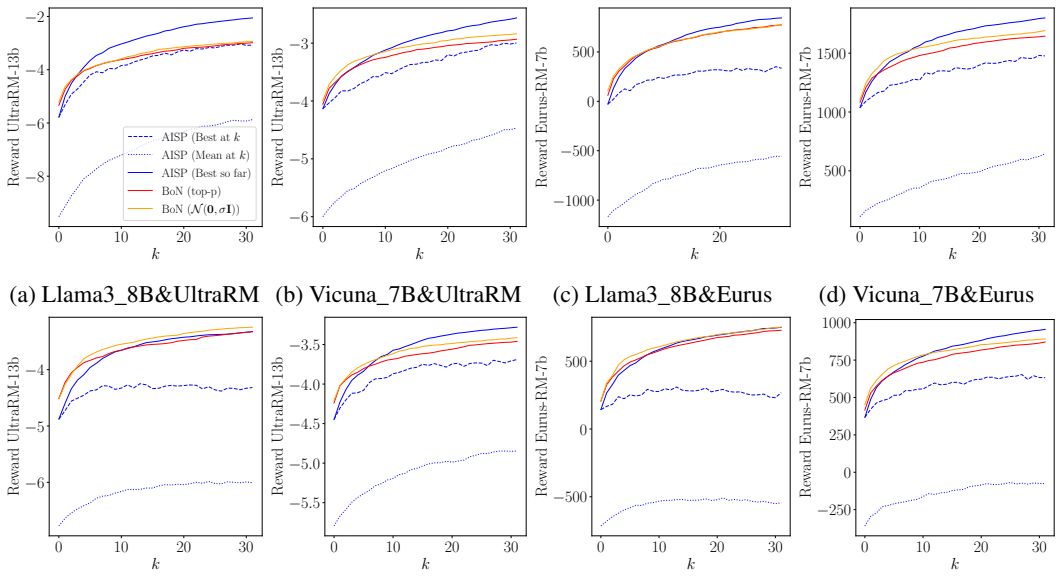
C.1 Hyperparameters

Hyper-parameter-tuning of AISP is performed on test data because our problems do not need to consider overfitting. For all conditions, we set $\alpha = 0.9999$ and $\sigma = 0.5$. We observe that AISP can stably optimize r when the standard deviation of $r(\mathbf{x}, \mathbf{y}(V^i))/\lambda$ is approximately one digit. We set $\lambda = 0.7$ for SHP with UltraRM and $\lambda = 0.3$ for HH-RLHF with UltraRM. and set $\lambda = 240$ for SHP with Eurus and $\lambda = 120$ for HH-RLHF with Eurus. n, τ, and $\kappa = 32$ are set to 32, 128, and 32, respectively.In BoN ($\mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$), σ^2 is set to 0.5.

C.2 Win rate

Tables 2 and 3 list the win rate for AISP vs BoN. To compute win rate, we sampled 100 pairs of prompts and responses at random, and GPT-4 judges whether the response from AISP or BoN is better. Since the values are averaged over three trials, they do not always sum to 100 %. These tables show that AISP has a high win rate against BoN (top-p) under all conditions, and a high win rate against BoN ($\mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$) under most conditions. The results of average rewards and win rates show that AISP aligns LLMs better than BoN when allowing sequential process.

Convergence Figure 2 plots curves of reward values during iterations on SHP and HH-RLHF, respectively. These curves are also evaluated on randomly selected 100 pairs, and are averaged over data samples and three trials. This figure shows that though AISP underperforms BoN in the early iterations, it improves more rapidly and eventually surpasses BoN as the number of iterations increases in most cases. Additionally, while the maximum number of iterations k was set to 32 in this experiment, it is likely that the performance gap would become more pronounced with a larger number of iterations. Reward values of AISP (Mean at k) and AISP (Best at k) also increase according to k. This indicates that AISP not only optimizes the resulting response but also optimizes the distribution of responses. Therefore, AISP obtains aligned distributions of response without any additional techniques.



(a) Llama3_8B&UltraRM (b) Vicuna_7B&UltraRM (c) Llama3_8B&Eurus (d) Vicuna_7B&Eurus

(e) Llama3_8B&UltraRM (f) Vicuna_7B&UltraRM (g) Llama3_8B&Eurus (h) Vicuna_7B&Eurus

Figure 2: Reward curve against iterations on SHP (top) and HH-RLHF (bottom). AISP (Mean at k) is $1/n \sum_i r(\mathbf{x}, \mathbf{y}(V^i))$. AISP (Best at k) is $\max_i r(\mathbf{x}, \mathbf{y}(V^i))$, and AISP (Best so far) is $\mathbf{y}_{\text{best}}$ in Algorithm 1 at k. BoN corresponds to $\max_{\mathbf{y} \in \mathcal{Y}_N} r(\mathbf{x}, \mathbf{y})$ using $N = nk$ samples where $n = 32$.