Rejection Sampling Based Fine Tuning Secretly Performs PPO

Gautham Govind Anil [1] **Dheeraj Nagaraj** [1] **Karthikeyan Shanmugam** [1] **Sanjay Shakkottai** [2]

Abstract

Several downstream applications of pre-trained generative models require task-specific adaptations based on reward feedback. In this work, we examine strategies to fine-tune a pre-trained model given non-differentiable rewards on generations. We establish connections between Rejection Sampling based fine-tuning and Proximal Policy Optimization (PPO) - we use this formalism to establish PPO with marginal KL constraints for diffusion models. A framework for intermediate denoising step fine-tuning is then proposed for more sample-efficient fine-tuning of diffusion models. Experimental results are presented on the tasks of layout generation and molecule generation to validate the claims.

1. Background

Recently, several works have demonstrated that rejection sampling based finetuning strategies perform on par with/ better than RL-based methods for both autoregressive language generation (Dong et al., 2023; Xiong et al., 2025) as well as diffusion-based image generation (Zhang et al., 2025). Further, it is also known that for diffusion models, where the marginal distribution is intractable, exact implementation of PPO (which would require computing marginal KL divergence) is not feasible (Fan et al., 2023) - rather, an approximation is done by using trajectory KL divergence instead.

In this work, we establish conceptual connections between Rejection Sampling based fine tuning and Proximal Policy Optimization (PPO) based reward maximization. Specifically, we make the following contributions:

1. For binary and continuous preference rewards, we establish that performing rejection sampling based fine tuning is equivalent to PPO with a reshaped reward and a fixed weight for KL regularization. In particular, this equivalence enables PPO with marginal KL constraint for diffusion models.

2. We then propose a sampling strategy to allow for finer KL control instead of a fixed KL penalty. We also propose a more general sampler which allows us to extend the formulation to general relative reward functions instead of just preference rewards.

3. We extend this framework by assigning the rewards of complete generations to partial generations. Specializing to the case of diffusion models, we propose a strategy which fine-tunes only until an intermediate denoising step - we examine how this affects the marginal distribution and discuss where this could be useful.

4. Experiments are reported on the tasks of layout generation and molecule generation. We empirically validate the benefit of marginal KL constraint for diffusion models and effectiveness of intermediate step rejection sampling based fine tuning.

Notation We use regular lower case letters (like x) to denote scalars, bold lower case letters (like $\mathbf{x}$) to denote vectors. We use regular upper case letters (like X) to denote random variables and bold upper case letters (like $\mathbf{X}$) to denote random vectors. The pre-trained/reference model (the model on which fine-tuning is to be done) will be denoted as $p^{\mathrm{r}}(\mathbf{x})$. Note that in general, the model may be sampling from a joint distribution of discrete and continuous variables - however, for the rest of the discussion, we assume $\mathbf{x} \in \mathbb{R}^d$ to be a continuous variable and $p^{\mathrm{r}}(\mathbf{x})$ to be a probability density function (the results extend to discrete/discrete-continuous cases as well).

2. Rejection Sampling - PPO Equivalence for Preference Rewards

Given a sample $\mathbf{x}$ sampled from the pre-trained model $p^{\mathrm{r}}(\mathbf{x})$, assume we have access to a scalar reward function $r : \mathbb{R}^d \to \mathbb{R}$. Then, the following holds:

Lemma 2.1. *Consider drawing two IID samples $\mathbf{X}_1$ and $\mathbf{X}_2$ according to the density $p^{r}(\mathbf{x})$ with corresponding rewards $r(\mathbf{X}_1)$ and $r(\mathbf{X}_2)$. Further, assume that $r(\mathbf{x})$ has a continuous probability density with CDF $F(r(\mathbf{x}))$. Then,*

[1]Google DeepMind [2]The University of Texas at Austin. Correspondence to: Gautham Govind Anil <gauthamga@google.com>.

the probability density of the higher reward samples:

$$p_{\mathbf{X}_2}\left(\mathbf{X}_2 = \mathbf{x} \middle| r(\mathbf{X}_2) \geq r(\mathbf{X}_1)\right)$$

is exactly the solution to the following optimization problem ($KL(\cdot\|\cdot)$ denotes the KL divergence):

$$\arg\max_{\hat{p}} \left[\mathbb{E}_{\mathbf{x}\sim\hat{p}(\cdot)}\hat{r}(\mathbf{x}) - \alpha KL\left(\hat{p}(\cdot)\|p^r(\cdot)\right)\right] \tag{1}$$

with $\frac{\hat{r}(\mathbf{x})}{\alpha} = \log F(r(\mathbf{x}))$.

Note that (1) is exactly the PPO optimization objective with $\hat{r}(\cdot)$ as the reward function, α as the KL regularization weight and $p^{\mathrm{r}}(\cdot)$ as the reference policy. Hence, sampling from the distribution learned by training on the higher reward samples is equivalent to sampling from the optimal policy of (reward-shaped) PPO objective.

In practice, the training algorithm is relatively straightforward: obtain sample pairs from the pre-trained model, filter them to retain only higher reward samples and finetune the model on these samples. The algorithm is described in Algorithm 1. Note that Algorithm 1 describes a sampling strategy which implicitly computes KL constraint always with the reference model $p^{\mathrm{r}}(\cdot)$. It is also possible to do multiple rounds of Algorithm 1 where the reference model is replaced by the current fine-tuned model after each round. This could result in better rewards, but would also cause the model to move farther away from the reference model.

Algorithm 1 Rejection Sampling Fine-Tuning

Input: Trainable sampler $p_\theta(\cdot)$, Reference sampler $p^{\mathrm{r}}(\cdot)$, Reward function $r(\cdot)$

1: Sample N pairs:
$\mathbf{X} = \{(X_1^{(1)}, X_1^{(2)}), \ldots, (X_N^{(1)}, X_N^{(2)})\}$ using $p^{\mathrm{r}}(\cdot)$.
2: Obtain rewards:
$\mathbf{R} = \left\{\left(r(X_1^{(1)}), r(X_1^{(2)})\right), \ldots, \left(r(X_N^{(1)}), r(X_N^{(2)})\right)\right\}$
3: Filter the pairs in $\mathbf{X}$ using $\mathbf{R}$ and retain only the higher reward samples to obtain $\tilde{\mathbf{X}}$.
4: Form a dataset $\mathcal{D}$ using the samples from $\tilde{\mathbf{X}}$.
5: Train p_θ on $\mathcal{D}_k$.

2.1. Formulation for Binary Rewards

Lemma 2.1 has been stated with the assumption of a continuous density for the rewards - however, it is possible to extend it to the case of binary rewards, i.e., $r : \mathbb{R}^d \to \{0, 1\}$ since the CDF $F(r(\mathbf{x}))$ is well-defined even for discrete rewards. For binary rewards, $r(\mathbf{X}_2) \geq r(\mathbf{X}_1)$ would imply selecting the higher reward sample when the rewards are different ($\{0, 1\}$ or $\{1, 0\}$) and one sample randomly when the rewards are equal ($\{0, 0\}$ or $\{1, 1\}$). Hence, the sampler would still sample 0 reward samples with a non-zero probability. It is possible to train only on samples with reward 1 by using the sampler: $p_{\mathbf{X}_2}\left(\mathbf{X}_2 = \mathbf{x} \middle| r(\mathbf{X}_2) = 1\right)$. This would tilt the reference distribution more since samples with reward 0 are no longer sampled - the exact tilted distribution is given in Appendix A.1.1. Training algorithm is even simpler in this case - obtain multiple samples from the pre-trained model, retain only samples with a reward of 1 and finetune the model on these samples.

2.2. Implications for Diffusion Models

While the discussion so far is generic, there are certain advantages of the rejection sampling formalism specific to diffusion models.

Recall that in diffusion models, we start from pure noise ($\mathbf{x}_T$) and use the learned diffusion model $p^r(\mathbf{x}_t|\mathbf{x}_{t+1})$ to obtain the sample $\mathbf{x}_0$ (Ho et al., 2020). Note that there is no closed form expression for $p^r(\mathbf{x}_0)$ - we can only obtain *samples* by following the reverse process. Due to this, it is *not possible* to compute the KL divergence between marginals as is required for PPO (see (1)), as noted in (Fan et al., 2023).

Existing works (Fan et al., 2023; Wallace et al., 2024) overcome this issue by optimizing a *trajectory* KL constraint instead of the *marginal* KL constraint, i.e., $KL\left(\hat{p}(\mathbf{x}_t|\mathbf{x}_{t+1})\|p^{\mathrm{r}}(\mathbf{x}_t|\mathbf{x}_{t+1})\right)$ (for all t) instead of $KL\left(\hat{p}(\mathbf{x}_0)\|p^{\mathrm{r}}(\mathbf{x}_0)\right)$. While it can be shown that that trajectory KL constraint is an upper bound of the marginal KL constraint, note that this also means that optimization is done over a more restrictive space.

Intuitively, it is possible that multiple trajectories could lead to the same final sample $\mathbf{x}_0$ - by imposing a trajectory KL constraint, the fine-tuning is limited to trajectories close to the reference trajectory. However, by relaxing this to a marginal KL constraint, it is possible to learn far away trajectories, provided the marginal distribution remains close.

While PPO with marginal constraints cannot be implemented for diffusion models explicitly, the equivalence in Lemma 1 can be used to cast this as a rejection sampling problem. Hence, by using the rejection sampling equivalence, *PPO with marginal constraints can be implemented implicitly using Algorithm 1, despite not having explicit access to the marginal distributions.*

3. Rejection Sampling Strategy for Finer KL Control

Lemma 2.1 establishes equivalence with PPO, but there is no explicit KL control like PPO since the ratio $\frac{\hat{r}(\mathbf{x})}{\alpha}$ is fixed. We now provide a sampler capable of providing finer control on the KL regularization:

Lemma 3.1. *Consider drawing two IID samples* $\mathbf{X}_1$ *and*

$\mathbf{X}_2$ *according to the density* $p^r(\mathbf{x})$ *with corresponding rewards* $r(\mathbf{X}_1)$ *and* $r(\mathbf{X}_2)$*. Further, assume that* $r(\mathbf{x})$ *has a continuous probability density with CDF* $F(r(\mathbf{x}))$*. Draw a sample* U *from the uniform distribution Unif*$[0,1]$*. Let* E_1 *denote the event* $r(\mathbf{X}_2) \geq r(\mathbf{X}_1)$ *and* E_2 *denote the event* $(U \leq \beta) \cap (r(X_2) < r(X_1))$ *(*$\beta \in [0,1]$*). If* $E = E_1 \cup E_2$*, the probability density:*

$$p_{\mathbf{X}_2}\left(\mathbf{X}_2 = \mathbf{x} \middle| E\right)$$

is exactly the solution to the following optimization problem (KL$(\cdot\|\cdot)$ *denotes the KL divergence):*

$$\arg\max_{\hat{p}} \left[\mathbb{E}_{\mathbf{x}\sim\hat{p}(\cdot)}\hat{r}(\mathbf{x}) - \alpha KL\left(\hat{p}(\cdot)\|p^r(\cdot)\right)\right] \tag{2}$$

with $\frac{\hat{r}(\mathbf{x})}{\alpha} = \log\left[\frac{2}{\beta}\left(\beta + (1-\beta)F(r(\mathbf{x}))\right)\right]$.

Lemma 3.1 allows us to control the tilt of the distribution more finely by varying the control parameter β. $\beta = 0$ recovers the result in Lemma 2.1, while $\beta = 1$ results in no tilt at all. From an algorithmic perspective, to implement this strategy, the only change required in Algorithm 1 is in step 3: with a probability β, the lower reward sample is also included during the filtration.

4. Rejection Sampling - PPO Equivalence for General Reward Functions

While Section 2 provides a rejection sampling based fine-tuning strategy for a preference-based reward comparison, in general, it might be necessary to use more general reward comparisons. For instance, it might be important to consider how better one sample is with respect to the other, i.e., $|r(\mathbf{X_1}) - r(\mathbf{X_2})|$, rather than just the ordering. Towards this, we present rejection sampling - PPO equivalence in scenarios with more general relative reward functions.

Given a sample $\mathbf{x}$ sampled from the pre-trained model $p^{\mathrm{r}}(\mathbf{x})$, assume we have access to a continuous scalar reward function $r : \mathbb{R}^d \to \mathbb{R}$.Suppose we are also given an arbitrary relative reward function $d(r(\mathbf{x}_1), r(\mathbf{x}_2))$ given two rewards. Without loss of generality, we assume $d(\cdot,\cdot)$ to be normalized to the range $[0,1]$. Then the following holds:

Lemma 4.1. *Consider drawing two IID samples* $\mathbf{X}_1$ *and* $\mathbf{X}_2$ *according to the density* $p^r(\mathbf{x})$ *with corresponding rewards* $r(\mathbf{X}_1)$ *and* $r(\mathbf{X}_2)$*. Without loss of generality, assume* $r(\mathbf{X}_2) \geq r(\mathbf{X}_1)$ *and let the relative reward be* $d(r(\mathbf{X}_1), r(\mathbf{X}_2))$*. Sample* $U \sim$ *Unif*$[0,1]$*. Then:*

$$p_{\mathbf{X}_2}\left(\mathbf{X}_2 = \mathbf{x} \middle| d(r(\mathbf{X}_2), r(\mathbf{X}_1)) > U\right)$$

is exactly the solution to the following optimization problem (KL$(\cdot\|\cdot)$ *denotes the KL divergence):*

$$\arg\max_{\hat{p}} \left[\mathbb{E}_{\mathbf{x}\sim\hat{p}(\cdot)}\left(\hat{r}(\mathbf{x})\right) - \alpha KL\left(\hat{p}(\cdot)\|p^r(\cdot)\right)\right] \tag{3}$$

with $\frac{\hat{r}(\mathbf{x})}{\alpha} = \log \mathbb{E}_{\mathbf{y}\sim p^r(\cdot)} d(r(\mathbf{x}), r(\mathbf{y}))$.

Intuitively, the reshaped reward function for PPO, $\hat{r}(\mathbf{X})$, looks at *how better sample* $\mathbf{X}$ *is on average with respect to the other samples according to the relative reward function* $d(\cdot,\cdot)$. The better the sample, the higher the reward assigned to that sample and consequently, higher the probability density assigned to that sample.

5. Intermediate Step Rejection Sampling for Diffusion Models

We now examine if it is possible to make the rejection sampling strategy more efficient, in particular for diffusion models. Fine-tuning on the higher rewards samples requires fine-tuning the pre-trained diffusion model across all diffusion time steps. This could be sample inefficient, especially considering the fact that score estimation is harder closer to the data distribution (Chen et al., 2023). As an attempt to mitigate this potential issue, we consider the possibility of fine-tuning the pre-trained model only until an intermediate denoising step instead of fine-tuning across all time steps.

Consider a pre-trained diffusion model with the (intractable) marginal density $p^{\mathrm{r}}(\mathbf{x}_0)$. Assume that the model has T denoising steps: $T-1, T-2, \ldots, 0$. For some intermediate denoising time step t, let us denote the marginal density of $\mathbf{x}_t$ as $\tilde{p}^{\mathrm{r}}(\mathbf{x}_t)$. Note that $\tilde{p}^{\mathrm{r}}(\cdot)$ is also intractable. We denote the conditional density at intermediate timestep t given a sample at intermediate timestep $t+k$ ($k > 0$) as $p^{\mathrm{r}}(\mathbf{x}_t|\mathbf{x}_{t+k})$. Note that this quantity is tractable. Again, assume a scalar reward function, $r : \mathbb{R}^d \to \mathbb{R}$. Note that reward is assigned only at timestep 0. Then:

Lemma 5.1. *Consider drawing two IID samples* $\tilde{\mathbf{X}}_1$ *and* $\tilde{\mathbf{X}}_2$ *from the density* $\tilde{p}^r(\mathbf{x}_t)$*. Further, samples* $\mathbf{X}_1$ *and* $\mathbf{X}_2$ *are drawn from* $p^r(\mathbf{x}_0|\mathbf{x}_t = \tilde{\mathbf{X}}_1)$ *and* $p^r(\mathbf{x}_0|\mathbf{x}_t = \tilde{\mathbf{X}}_2)$ *respectively. Denote by* $\tilde{p}(\mathbf{x})$ *the density* $p_{\tilde{\mathbf{X}}_2}(\tilde{\mathbf{X}}_2 = \mathbf{x}|r(\mathbf{X}_2) \geq r(\mathbf{X}_1))$*. If* $\mathbf{x}_t$ *is sampled according to* $\tilde{p}(\mathbf{x})$*, the new tilted marginal density for* $\mathbf{x}_0$*, denoted as* $p(\cdot)$ *is given by:*

$$p(\mathbf{x}) = \int_{\tilde{\mathbf{x}}} p^r(\mathbf{x}|\tilde{\mathbf{x}})\tilde{p}(\tilde{\mathbf{x}})d\tilde{\mathbf{x}} \tag{4}$$

Further $p(\cdot)$ *has the following properties (*$I(\cdot;\cdot)$ *denotes mutual information and* $H(\cdot)$ *denotes entropy):*

- *If* $I(r(\mathbf{X}_2); \tilde{\mathbf{X}}_2) = 0$*,* $p(\mathbf{x}) = p^r(\mathbf{x})$*.*
- *If* $I(r(\mathbf{X}_2); \tilde{\mathbf{X}}_2) = H(r(\mathbf{X}_2))$*,* $p(\mathbf{x})$ *is the optimal solution to* (1)*.*

From Lemma 5.1, it is clear that the *mutual information* $I(R; \tilde{\mathbf{X}})$- between random variables R and $\tilde{\mathbf{X}}$ sampled from densities $p^{\mathrm{r}}(r(\mathbf{x}_0))$ and $\tilde{p}^{\mathrm{r}}(\mathbf{x}_t)$ respectively - determines the final tilt of the probability distribution.

Intuitively, fine-tuning until an intermediate state is effective only if there is *sufficient distinguishability* between intermediate states which lead to high reward samples and intermediate states which lead to low reward samples. If this distinguishability exists, the model can be fine-tuned to assign more weight to intermediate states which lead to high reward samples.

The fine-tuning algorithm is given in Appendix B. Note that the fine-tuned model is used for denoising only until the intermediate denoising timestep - the pre-trained model is used for further denoising.

6. Experiments

6.1. Tasks and Pre-trained Models

All experiments are done on pre-trained models trained using the IGD framework (Anil et al., 2025). IGD is a diffusion framework capable of handling both discrete and continuous data. The rejection sampling strategy is well-suited to the discrete-continuous setting, since apart from benefits stated already, implementing PPO would also require careful balancing of losses corresponding to discrete and continuous variables, as well as their corresponding regularizers for stable training. More details regarding the IGD framework can be found in (Anil et al., 2025) - however, for the rest of the discussion, it suffices to think of IGD as being similar to the standard DDPM setting (Ho et al., 2020).

We conduct experiments for two tasks: (i) layout generation, and (ii) molecule generation. Both these tasks involve generation with elements having discrete and continuous variables. The task of layout generation is to compose N elements $\{e_i\}_{i=1}^N$, where each element e_i corresponds to a category type $t_i \in \mathbb{N}$ along with a spatial position encoded through a bounding box vector $\mathbf{p}_i \in \mathbb{R}^4$. The other task is that of molecule generation, where a molecule is represented by a sequence of (atom, spatial location) pairs $(z_i, \mathbf{p}_i)_{i=1}^n$, where $z_i \in \mathbb{N}$ is the atomic number of the i-th element and $\mathbf{p}_i \in \mathbb{R}^3$ is its corresponding spatial position. We use PubLayNet (Zhong et al., 2019) for layout generation, and QM9 (Ramakrishnan et al., 2014) for molecule generation.

6.2. Marginal KL v/s Trajectory KL

To examine the effect of optimizing the marginal KL (as per Lemma 2.1) as opposed to trajectory KL, we first fine-tune the pre-trained model on high reward samples without any explicit KL constraint and then repeat this experiment but with an explicit trajectory KL constraint. We consider the task of layout generation with 'overlap' as the reward metric. We assign a reward of 1 to generations with 0 overlap and a reward of 0 otherwise. Further, we consider the tasks of category conditioning and category + size conditioning, where the categories and categories + sizes of the elements respectively are fixed. The results are given in Table 1.

Table 1. Layout Generation: Fine-tuning results for category-conditioned (C) and category+size-conditioned (CS)

	C		C+S	
Model	Overlap	FID	Overlap	FID
Baseline	0.013	4.07	0.027	0.886
Marginal KL	**0.006**	5.04	**0.017**	1.287
Trajectory KL	0.010	4.08	0.025	0.909

For the same number of training steps, it can be seen that training with marginal KL constraint allows the model to improve much faster on the overlap metric as opposed to trajectory KL. Further, the FID score remains close to the baseline model. This implies that the model is able to still produce fairly diverse samples, confirming that the model does not collapse to only a few samples.

6.3. Effectiveness of Intermediate Time Step Rejection Sampling

We implement the intermediate step rejection sampling strategy as described in Section 5 in the task of Molecule Generation. Given a pre-trained model, we assign a reward of 1 to generated molecules which are stable and 0 to the rest. If the pre-trained model is trained for T timesteps, we examine the effect of fine-tuning the first $T/4, T/2, 3T/4$ and T (complete fine-tuning) timesteps. We again compare with the complete fine-tuning with explicit trajectory KL as a baseline. The results are reported in Table 2.

The best performance is obtained by doing fine-tuning for the first 3 rounds. For less number of rounds, as discussed in Section 5, the distinguishability is less. Complete fine-tuning on the other hand is very sample inefficient since the denoiser has to be moved for all denoising steps - this is evident from the high sampling cost for complete fine tuning. Trajectory KL forces the model to stay close to baseline despite being trained for more number of steps. Hence, intermediate time step learning strategy is effective, provided we select a time step with sufficient distinguishability.

Table 2. Molecule Generation: Fine-tuning (FT) results for different timesteps. (Relative) number of sampling steps required are also reported.

Model	Mol: Stability	Uniqueness	Sampling Steps
Baseline	84.00	90.89	-
Trajectory KL	86.06	90.63	7x
$T/4$ steps FT	83.94	90.80	1x
$T/2$ steps FT	84.57	90.81	1x
$3T/4$ steps FT	**88.36**	88.45	1x
T steps FT	87.13	91.27	9x

References

Anil, G. G., Yadav, S., Nagaraj, D., Shanmugam, K., and Jain, P. Interleaved gibbs diffusion for constrained generation. arXiv preprint arXiv:2502.13450, 2025.

Chen, M., Huang, K., Zhao, T., and Wang, M. Score approximation, estimation and distribution recovery of diffusion models on low-dimensional data. In International Conference on Machine Learning, pp. 4672–4712. PMLR, 2023.

Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., Shum, K., and Zhang, T. Raft: Reward ranked finetuning for generative foundation model alignment. arXiv preprint arXiv:2304.06767, 2023.

Fan, Y., Watkins, O., Du, Y., Liu, H., Ryu, M., Boutilier, C., Abbeel, P., Ghavamzadeh, M., Lee, K., and Lee, K. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. Advances in Neural Information Processing Systems, 36:79858–79885, 2023.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. Advances in neural information processing systems, 33:6840–6851, 2020.

Ramakrishnan, R., Dral, P. O., Rupp, M., and Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. Scientific data, 1(1):1–7, 2014.

Wallace, B., Dang, M., Rafailov, R., Zhou, L., Lou, A., Purushwalkam, S., Ermon, S., Xiong, C., Joty, S., and Naik, N. Diffusion model alignment using direct preference optimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8228–8238, 2024.

Xiong, W., Yao, J., Xu, Y., Pang, B., Wang, L., Sahoo, D., Li, J., Jiang, N., Zhang, T., Xiong, C., et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. arXiv preprint arXiv:2504.11343, 2025.

Zhang, X., Wei, X., Wu, J., Wu, J., Zhang, Z., Lei, Z., and Li, Q. Generating on generated: An approach towards self-evolving diffusion models. arXiv preprint arXiv:2502.09963, 2025.

Zhong, X., Tang, J., and Yepes, A. J. Publaynet: largest dataset ever for document layout analysis. In 2019 International Conference on Document Analysis and Recognition (ICDAR), 2019. doi: 10.1109/ICDAR.2019.00166.

A. Proofs

A.1. Lemma 2.1

Consider drawing two IID samples $\mathbf{X}_1$ and $\mathbf{X}_2$ according to the density $p^{\mathrm{r}}(\mathbf{x})$ with corresponding rewards $r(\mathbf{X}_1)$ and $r(\mathbf{X}_2)$. Further, assume that $r(\mathbf{x})$ has a continuous probability density with CDF $F(r(\mathbf{x}))$. Then, the probability density of the higher reward samples:

$$p_{\mathbf{X}_2}\left(\mathbf{X}_2 = \mathbf{x} \middle| r(\mathbf{X}_2) \geq r(\mathbf{X}_1)\right)$$

is exactly the solution to the following optimization problem ($\mathrm{KL}(\cdot\|\cdot)$ denotes the KL divergence):

$$\arg\max_{\hat{p}} \left[\mathbb{E}_{\mathbf{x}\sim\hat{p}(\cdot)}\hat{r}(\mathbf{x}) - \alpha \mathrm{KL}\left(\hat{p}(\cdot)\|p^{\mathrm{r}}(\cdot)\right)\right] \tag{5}$$

with $\frac{\hat{r}(\mathbf{x})}{\alpha} = \log F(r(\mathbf{x}))$.

Proof. Consider the following probability measure(where A is any measurable set):

$$\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) \geq r(\mathbf{X}_1)) \tag{6}$$

Since $\mathbf{X}_1$ and $\mathbf{X}_2$ are IID samples, using Bayes' theorem, we have:

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) \geq r(\mathbf{X}_1)) &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, r(\mathbf{X}_2) \geq r(\mathbf{X}_1))}{\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1))} \\ &= 2\mathbb{P}(\mathbf{X}_2 \in A, r(\mathbf{X}_2) \geq r(\mathbf{X}_1))\end{aligned}$$

Therefore, we have ($\mathbf{1}$ denotes the indicator function):

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) \geq r(\mathbf{X}_1)) &= 2\int_{\mathbf{x}\in A}\left(\int_{\mathbf{y}} \mathbf{1}_{r(\mathbf{x})\geq r(\mathbf{y})} p^{\mathrm{r}}(\mathbf{y}) d\mathbf{y}\right) p^{\mathrm{r}}(\mathbf{x}) d\mathbf{x} \\ &= \int_{\mathbf{x}\in A} 2F_p(r(\mathbf{x}))p^{\mathrm{r}}(\mathbf{x}) d\mathbf{x}\end{aligned}$$

where $F(r(\mathbf{x}))$ denotes the CDF of $r(\mathbf{x})$. Hence, from the Radon-Nikodym theorem, we can claim that sampling according to the probability distribution $\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) \geq r(\mathbf{X}_1))$ is equivalent to sampling from the probability density $\tilde{p}(x) = 2F(r(x))p^{\mathrm{r}}(x)$. Note that we can rewrite this as:

$$\begin{aligned}&\tilde{p}(\mathbf{x}) = \exp\{[\log(2F(r(\mathbf{x})))]\}p^{\mathrm{r}}(\mathbf{x}) \\ \implies &\tilde{p}(\mathbf{x}) = 2\exp\{[\log(F(r(\mathbf{x})))]\}p^{\mathrm{r}}(\mathbf{x})\end{aligned} \tag{7}$$

Now consider the Proximal Policy Optimization (PPO) optimization objective (where $\hat{r}(\cdot)$ is some reward function):

$$p^{(\alpha)}(\cdot) = \arg\max_{\hat{p}} \left[\mathbb{E}_{\mathbf{x}\sim\hat{p}}\left[\hat{r}(\mathbf{x})\right] - \alpha\left(\mathrm{KL}\left(\hat{p}(\cdot)||p^{\mathrm{r}}(\cdot)\right)\right)\right]$$

The optimal solution is clearly:

$$p^{(\alpha)}(\mathbf{x}) = C\exp(\hat{r}(\mathbf{x})/\alpha)p^{\mathrm{r}}(\mathbf{x}) \tag{8}$$

where C is a normalizing constant independent of $\mathbf{x}$. Comparing (7) and (8), we have (with $C = 2$):

$$\hat{r}(\mathbf{x})/\alpha = \log(F(r(\mathbf{x})))$$

□

A.1.1. BINARY REWARDS

Assume $\mathbb{P}(r(\mathbf{x}) = 1)$ is a (where $\mathbf{x}$ is sampled from $p^{\mathrm{r}}(\cdot)$). Then, since we have binary rewards, $\mathbb{P}(r(\mathbf{x}) = 0) = 1 - a$. Therefore:

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) \geq r(\mathbf{X}_1)) &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, r(\mathbf{X}_2) \geq r(\mathbf{X}_1))}{\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1))} \\ &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, r(\mathbf{X}_2) \geq r(\mathbf{X}_1))}{a^2 + a(1-a) + (1-a)^2}\end{aligned}$$

and hence $C = \frac{1}{a^2 + a(1-a) + (1-a)^2}$. Further the CDF is now:

$$\begin{aligned}F(r(\mathbf{x}) = 0) &= 1 - a \\ F(r(\mathbf{x}) = 0) &= 1\end{aligned}$$

The proof for continuous rewards directly carries over for the case $\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) \geq r(\mathbf{X}_1))$ with the above-mentioned C and $F(r(\mathbf{x}))$. Now consider:

$$\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) = 1)$$

Using Bayes' theorem, we have:

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) = 1) &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, r(\mathbf{X}_2) = 1)}{\mathbb{P}(r(\mathbf{X}_2) = 1)} \\ &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, r(\mathbf{X}_2) = 1)}{a}\end{aligned}$$

Hence, we have the following shift:

$$p^{\alpha}(\mathbf{x}) = \begin{cases} \frac{p^{\mathrm{r}}(\mathbf{x})}{a}, & \text{if } r(\mathbf{x}) = 1 \\ 0, & \text{otherwise} \end{cases}$$

Compared to the case of $\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) \geq r(\mathbf{X}_1))$, $\mathbb{P}(\mathbf{X}_2 \in A | r(\mathbf{X}_2) = 1)$ tilts more since all the 0 reward samples are now assigned a probability of 0.

A.2. Lemma 3.1

Consider drawing two IID samples $\mathbf{X}_1$ and $\mathbf{X}_2$ according to the density $p^{\mathrm{r}}(\mathbf{x})$ with corresponding rewards $r(\mathbf{X}_1)$ and $r(\mathbf{X}_2)$. Further, assume that $r(\mathbf{x})$ has a continuous probability density with CDF $F(r(\mathbf{x}))$. Draw a sample U from the uniform distribution Unif$[0, 1]$. Let E_1 denote the event $r(\mathbf{X}_2) \geq r(\mathbf{X}_1)$ and E_2 denote the event $(U \leq \beta) \cap (r(X_2) < r(X_1))$ ($\beta \in [0, 1]$). If $E = E_1 \cup E_2$, the probability density:

$$p_{\mathbf{X}_2}\left(\mathbf{X}_2 = \mathbf{x} \middle| E\right)$$

is exactly the solution to the following optimization problem (KL$(\cdot \| \cdot)$ denotes the KL divergence):

$$\arg\max_{\hat{p}} \left[\mathbb{E}_{\mathbf{x} \sim \hat{p}(\cdot)} \hat{r}(\mathbf{x}) - \alpha \mathrm{KL}\left(\hat{p}(\cdot) \| p^{\mathrm{r}}(\cdot)\right)\right] \tag{9}$$

with $\frac{\hat{r}(\mathbf{x})}{\alpha} = \log\left[\frac{2}{\beta}\left(\beta + (1-\beta)F(r(\mathbf{x}))\right)\right]$.

Proof. From Bayes' theorem:

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A | E) &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, E)}{\mathbb{P}(E)} \\ &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, E)}{\mathbb{P}(E_1) + \mathbb{P}(E_2)} \\ &= \frac{2}{1+\beta}\mathbb{P}(\mathbf{X}_2 \in A, E)\end{aligned}$$

Therefore, we have ($\mathbf{1}$ denotes the indicator function):

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A|E) &= \frac{2}{1+\beta}\int_{\mathbf{x}\in A}\left(\int_{\mathbf{y}}\left(\mathbf{1}_{r(\mathbf{x})\geq r(\mathbf{y})}+\beta\mathbf{1}_{r(\mathbf{x})<r(\mathbf{y})}\right)p^{\mathrm{r}}(\mathbf{y})dy\right)p^{\mathrm{r}}(\mathbf{x})d\mathbf{x}\\ &= \int_{\mathbf{x}\in A}\frac{2}{1+\beta}\left[\beta+(1-\beta)F(r(\mathbf{x}))\right]p^{\mathrm{r}}(\mathbf{x})d\mathbf{x}\end{aligned}$$

Hence, we have:

$$\hat{p}(\mathbf{x}) = \frac{2}{1+\beta}\left[\beta+(1-\beta)F(r(\mathbf{x}))\right]p^{\mathrm{r}}(\mathbf{x})$$

For $\beta = 0$, we recover the density in Lemma 2.1, while for $\beta = 1$, we have $\hat{p} = p^{\mathrm{r}}$. Hence, β can be used as a slider to control the KL constraint. □

A.3. Lemma 4.1

Consider drawing two IID samples $\mathbf{X}_1$ and $\mathbf{X}_2$ according to the density $p^{\mathrm{r}}(\mathbf{x})$ with corresponding rewards $r(\mathbf{X}_1)$ and $r(\mathbf{X}_2)$. Without loss of generality, assume $r(\mathbf{X}_2) \geq r(\mathbf{X}_1)$ and let the relative reward be $d(r(\mathbf{X}_1), r(\mathbf{X}_2))$. Sample $U \sim \text{Unif}[0,1]$. Then, the density:

$$p_{\mathbf{X}_2}\left(\mathbf{X}_2 = \mathbf{x}\middle| d(r(\mathbf{X}_2), r(\mathbf{X}_1)) > U\right)$$

is exactly the solution to the following optimization problem ($\mathrm{KL}(\cdot\|\cdot)$ denotes the KL divergence):

$$\arg\max_{\hat{p}}\left[\mathbb{E}_{\mathbf{x}\sim\hat{p}(\cdot)}\left(\hat{r}(\mathbf{x})\right) - \alpha\mathrm{KL}\left(\hat{p}(\cdot)\|p^{\mathrm{r}}(\cdot)\right)\right] \tag{10}$$

with $\frac{\hat{r}(\mathbf{x})}{\alpha} = \log\mathbb{E}_{\mathbf{y}\sim p^{\mathrm{r}}(\cdot)}d(r(\mathbf{x}), r(\mathbf{y}))$.

Proof. Consider the following probability measure(where A is any measurable set):

$$\mathbb{P}(\mathbf{X}_2 \in A|d(r(\mathbf{X}_2), r(\mathbf{X}_1)) > U) \tag{11}$$

where $d(r(\mathbf{X}_2), r(\mathbf{X}_1))$ is some distance computed between rewards $r(\mathbf{X}_2)$ and $r(\mathbf{X}_1)$ and $U \sim \text{Unif}[0,1]$. For ease of notation, we denote $d(r(\mathbf{X}_2), r(\mathbf{X}_1))$ as $d(\mathbf{X}_2, \mathbf{X}_1)$ for further discussion. Further, assume $d(\mathbf{X}_2, \mathbf{X}_1) \in [0,1]$ (this can be achieved by normalizing the rewards). Now, from Bayes' theorem, we have:

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A|d(\mathbf{X}_2, \mathbf{X}_1) > U) &= \frac{\mathbb{P}(\mathbf{X}_2 \in A, d(\mathbf{X}_2, \mathbf{X}_1))}{\mathbb{P}(d(\mathbf{X}_2, \mathbf{X}_1) > U)}\\ &= Z\mathbb{P}(\mathbf{X}_2 \in A, d(\mathbf{X}_2, \mathbf{X}_1) > U)\end{aligned}$$

for some normalizing constant $Z = \mathbb{E}_{\mathbf{x}_1,\mathbf{x}_2\sim p^{\mathrm{r}}(\cdot)}[d(\mathbf{x}_2, \mathbf{x}_1)]$ since $U \sim \text{Uni}[0,1]$. Now, we have:

$$\begin{aligned}\mathbb{P}(\mathbf{X}_2 \in A, d(\mathbf{X}_2, \mathbf{X}_1) > U) &= \int_{\mathbf{x}\in A}\left(\int_{\mathbf{y}}\left(\int_0^1 \mathbf{1}_{d(\mathbf{x},\mathbf{y})>u}du\right)p^{\mathrm{r}}(\mathbf{y})d\mathbf{y}\right)p^{\mathrm{r}}(\mathbf{x})d\mathbf{x}\\ &\overset{(a)}{=} \int_{\mathbf{x}\in A}\left(\int_{\mathbf{y}} d(\mathbf{x},\mathbf{y})p^{\mathrm{r}}(\mathbf{y})d\mathbf{y}\right)p^{\mathrm{r}}(\mathbf{x})d\mathbf{x}\\ &= \int_{\mathbf{x}\in A}\mathbb{E}_{\mathbf{y}\sim p^{\mathrm{r}}(\cdot)}[d(\mathbf{x},\mathbf{y})]p^{\mathrm{r}}(\mathbf{x})d\mathbf{x}\end{aligned}$$

where (a) follows from that fact that $u \sim \text{Unif}[0,1]$. Now, again from the Radon-Nikodym Theorem, sampling from $\mathbb{P}(\mathbf{X}_2 \in A|d(\mathbf{X}_2, \mathbf{X}_1) > U)$ is equivalent to sampling according to the density $\tilde{p}(\cdot)$:

$$\tilde{p}(\mathbf{x}) = Z\exp\left\{\left[\log\left(\mathbb{E}_{\mathbf{y}\sim p(\cdot)}[d(\mathbf{x},\mathbf{y})]\right)\right]\right\}p^{\mathrm{r}}(\mathbf{x})$$

Hence, this is equivalent to PPO with:

$$\hat{r}(\mathbf{x})/\alpha = \log\left(\mathbb{E}_{\mathbf{y}\sim p^{\mathrm{r}}(\cdot)}[d(\mathbf{x},\mathbf{y})]\right)$$

□

A.4. Lemma 5.1

Consider drawing two IID samples $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$ from the density $\tilde{p}^{\mathrm{r}}(\mathbf{x}_t)$. Further, samples $\mathbf{X}_1$ and $\mathbf{X}_2$ are drawn from $p^{\mathrm{r}}(\mathbf{x}_0|\mathbf{x}_t = \tilde{\mathbf{X}}_1)$ and $p^{\mathrm{r}}(\mathbf{x}_0|\mathbf{x}_t = \tilde{\mathbf{X}}_2)$ respectively. Denote by $\tilde{p}(\mathbf{x})$ the density:

$$p_{\tilde{\mathbf{X}}_2}(\tilde{\mathbf{X}}_2 = \mathbf{x}|r(\mathbf{X}_2) \geq r(\mathbf{X}_1))$$

If $\mathbf{x}_t$ is sampled according to $\tilde{p}(\mathbf{x})$, the new tilted marginal density for $\mathbf{x}_0$, denoted as $p(\cdot)$ is given by:

$$p(\mathbf{x}) = \int_{\tilde{\mathbf{x}}} p^{\mathrm{r}}(\mathbf{x}|\tilde{\mathbf{x}})\tilde{p}(\tilde{\mathbf{x}})d\tilde{\mathbf{x}} \tag{12}$$

Further $p(\cdot)$ has the following properties ($I(\cdot;\cdot)$ denotes mutual information and $H(\cdot)$ denotes entropy):

- If $I(r(\mathbf{X}_2); \tilde{\mathbf{X}}_2) = 0$, $p(\mathbf{x}) = p^{\mathrm{r}}(\mathbf{x})$.
- If $I(r(\mathbf{X}_2); \tilde{\mathbf{X}}_2) = H(r(\mathbf{X}_2))$, $p(\mathbf{x})$ is the optimal solution to (1).

Proof. We use Bayes' theorem:

$$\begin{aligned}
\mathbb{P}(\tilde{\mathbf{X}}_2 = \mathbf{x}|r(\mathbf{X}_2) \geq r(\mathbf{X}_1)) &= \frac{\mathbb{P}(\tilde{\mathbf{X}}_2 = \mathbf{x}, r(\mathbf{X}_2) \geq r(\mathbf{X}_1))}{\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1))} \\
&= \frac{\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1)|\tilde{\mathbf{X}}_2 = \mathbf{x})\mathbb{P}(\tilde{\mathbf{X}}_2 = \mathbf{x})}{\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1))} \\
&= 2\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1)|\tilde{\mathbf{X}}_2 = \mathbf{x})\mathbb{P}(\tilde{\mathbf{X}}_2 = \mathbf{x})
\end{aligned}$$

since if $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$ are IID, $\mathbf{X}_1$ and $\mathbf{X}_2$ are also IID. Hence, the new tilted distribution can now be written as:

$$p(\mathbf{x}) = 2\int_{\tilde{\mathbf{x}}} p^{\mathrm{r}}(\mathbf{x}|\tilde{\mathbf{x}})\mathbb{P}(r(\mathbf{X}_2 = \mathbf{x}) \geq r(\mathbf{X}_1)|\tilde{\mathbf{X}}_2 = \tilde{\mathbf{x}})\mathbb{P}(\tilde{\mathbf{X}}_2 = \tilde{\mathbf{x}})d\tilde{\mathbf{x}}$$

□

Case 1: $I(r(\mathbf{X}_2); \tilde{\mathbf{X}}_2) = 0$
Clearly $r(\mathbf{X}_2)r(\mathbf{X}_2)$ is independent of $\tilde{\mathbf{X}}_2$. Hence:

$$\begin{aligned}
p(\mathbf{x}) &= 2\int_{\tilde{\mathbf{x}}} p^{\mathrm{r}}(\mathbf{x}|\tilde{\mathbf{x}})\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1)|\tilde{\mathbf{X}}_2 = \tilde{\mathbf{x}})\mathbb{P}(\tilde{\mathbf{X}}_2 = \tilde{\mathbf{x}})d\tilde{\mathbf{x}} \\
&= 2\int_{\tilde{\mathbf{x}}} p^{\mathrm{r}}(\mathbf{x}|\tilde{\mathbf{x}})\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1))\mathbb{P}(\tilde{\mathbf{X}}_2 = \tilde{\mathbf{x}})d\tilde{\mathbf{x}} \\
&= p^{\mathrm{r}}(\mathbf{x})
\end{aligned}$$

Case 2: $I(r(\mathbf{X}_2); \tilde{\mathbf{X}}_2) = H(r(\mathbf{X}_2))$
$r(\mathbf{X}_2)$ is a deterministic function of $\tilde{\mathbf{X}}_2$, i.e., $r(\mathbf{X}_2) = \tilde{r}(\tilde{\mathbf{X}}_2)$ and $\mathbb{P}(r(\mathbf{X}_2)|\tilde{\mathbf{X}}_2) = \mathbf{1}_{r(\mathbf{X}_2)=\tilde{r}(\tilde{\mathbf{X}}_2)}$. Therefore, $\mathbb{P}(r(\mathbf{X}_2) \geq r(\mathbf{X}_1)|\tilde{\mathbf{X}}_2 = \tilde{\mathbf{x}}) = F_r(\tilde{r}(\tilde{\mathbf{x}}))$, where $F_r(\cdot)$ denotes the CDF of $r(\mathbf{X}_1)$, with $\mathbf{X}_1 \sim p^{\mathrm{r}}(\cdot)$. Note that since $r(\mathbf{X})$ is a deterministic function of $\tilde{\mathbf{X}}$, $F_r(\tilde{r}(\tilde{\mathbf{x}})) = F_r(r(\mathbf{x}))$, where $\mathbf{x}$ is the final sample obtained by denoising from $\tilde{\mathbf{x}}$. Hence, we have:

$$\begin{aligned}
p(\mathbf{x}) &= 2\int_{\tilde{\mathbf{x}}} p^{\mathrm{r}}(\mathbf{x}|\tilde{\mathbf{x}})F_r(r(\mathbf{x}))\mathbb{P}(\tilde{\mathbf{X}}_2 = \tilde{\mathbf{x}})d\tilde{\mathbf{x}} \\
&= 2F(r(\mathbf{x}))p^{\mathrm{r}}(\mathbf{x})
\end{aligned}$$

From the proof of Lemma 2.1, it can be seen that this is equivalent to the optimum of fine-tuning across all timesteps.

B. Intermediate Timestep Rejection Sampling

B.1. Algorithm

Algorithm 2 Intermediate Step Rejection Sampling FT

Input: Trainable diffusion model $p_\theta(\cdot)$, Pre-trained diffusion model $p^{\text{r}}(\cdot)$, Reward function $r(\cdot)$, Total denoising timesteps T, Intermediate denoising timestep T-K.

1: Sample N pairs by complete denoising using $p^{\text{r}}(\cdot)$ ($\mathbf{x}_0$):
$\mathbf{X} = \{(X_1^{(1)}, X_1^{(2)}), \ldots, (X_N^{(1)}, X_N^{(2)})\}$
and also collect the corresponding denoised states at timestep $T-K$ ($\mathbf{x}_{T-K}$):
$\tilde{\mathbf{X}} = \{(\tilde{X}_1^{(1)}, \tilde{X}_1^{(2)}), \ldots, (\tilde{X}_N^{(1)}, \tilde{X}_N^{(2)})\}$.
2: Obtain rewards:
$\mathbf{R} = \left\{ \left(r(X_1^{(1)}), r(X_1^{(2)}) \right), \ldots, \left(r(X_N^{(1)}), r(X_N^{(2)}) \right) \right\}$
3: Filter the pairs in $\tilde{\mathbf{X}}$ using $\mathbf{R}$ and retain only the higher reward samples to obtain $\hat{\tilde{\mathbf{X}}}$.
4: Form a dataset $\mathcal{D}$ using the samples from $\hat{\tilde{\mathbf{X}}}$.
5: Train p_θ on $\mathcal{D}_k$ for the denoising timesteps $\{T-1, T-2, \ldots, T-K\}$.

B.2. Intuition

The intuition behind the intermediate denoising step rejection sampling strategy can be seen clearly from the plots presented in Figure 1. The three plots demonstrate the distinguishability of intermediate states based on final rewards across different denoising timesteps for molecule generation. Note that for molecule generation, a reward of 1 corresponds to a stable molecule and a reward of 0 corresponds to an unstable molecule. X-axis represents the average reward obtained assuming a fixed intermediate denoising timestep across 100 samples. Y-axis represents the fraction of total intermediate timesteps which lead to a particular average reward - the fraction is computed over 1000 possible values of intermediate states. Essentially, we roll out until an intermediate state, and calculate the average reward corresponding to this intermediate state across 100 generations. We repeat this process 1000 times and plot the values. The experiment done after $T/4$ denoising timesteps, $T/2$ denoising timesteps and $3T/4$ denoising timesteps are represented in (a), (b) and (c) respectively of Figure 1. In each case, we also plot corresponding values for a Bernoulli random variable with probability of success equal to average molecule stability across all intermediate states - essentially this indicates how the distribution would look like if the intermediate state had no information at all regarding the final rewards. As can be seen from Figure 1, at $T/4$ timesteps, the two distributions are pretty much indistinguishable, indicating that the intermediate state is not informative of the final rewards at all. The distinguishability improves for $T/2$ timesteps and at $3T/4$ timesteps, it can be seem that with high probability, the intermediate state determines the final reward.

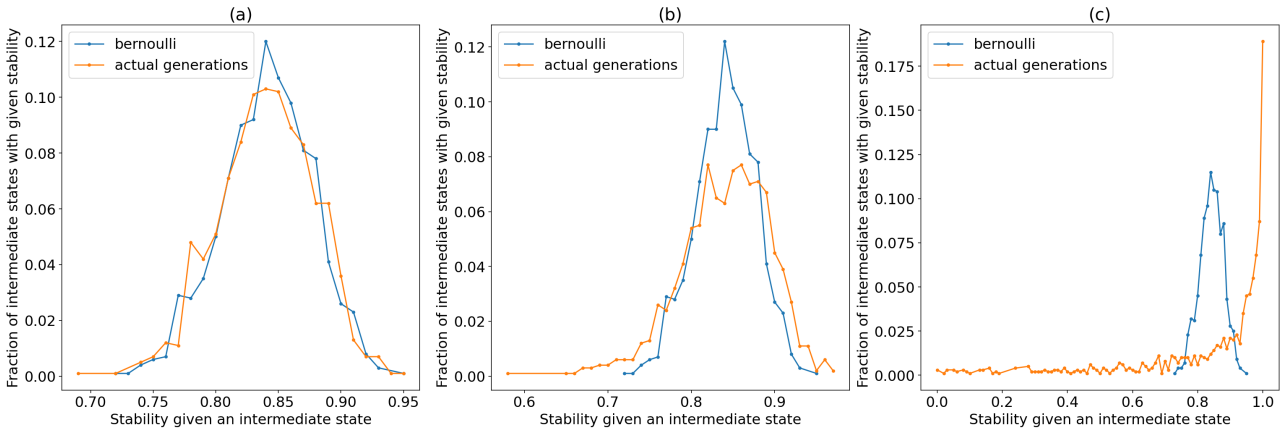

Figure 1. Distinguishability of intermediate states based on final rewards for molecule generation.

C. Experimental Details

For both layout generation and molecule generation, we use the same data pre-processing, architecture and hyperparameters as used in (Anil et al., 2025) to obtain the pre-trained models - further, the same architecture and hyperparameters are used for fine-tuning as well. Molecule generation results are reported on 32768 samples and without the inference-time refinement strategy ReDeNoise implemented in (Anil et al., 2025) - hence the reported value for the baseline model are lower (effect of ReDeNoise is discussed in Appendix D). For both tasks, we generate 32768 samples in every sampling step - the models are fine-tuned on these samples for 10000 steps using a training batch size of 4096. For layout generation, training is done for 20 sampling steps for both marginal KL and trajectory KL versions. For molecule generation, intermediate denoising step training requires 100 sampling steps, the trajectory KL version requires 700 steps and fine-tuning across all timesteps requires 900 sampling steps. For both tasks, fine-tuning is done only on high reward samples, samples corresponding to reward 0 are discarded. For trajectory KL, the KL weight α is set to 1 for both tasks.

D. Additional Experiments

D.1. Intermediate Time Step Fine-Tuning for Layout Generation

The intermediate time step rejection sampling fine-tuning was also tested out for layout generation - the results are given in Table 3.

	C	
Model	Overlap	FID
Baseline	0.013	4.07
Trajectory KL	0.010	4.08
T steps FT	**0.006**	5.04
$3T/4$ steps FT	0.008	4.12

Table 3. Additional Fine-tuning results for layout generation

It can be seen that fine-tuning until an intermediate step is more effective than imposing trajectory KL but not as effective as fine-tuning across all timesteps. This points to a bias-variance tradeoff: while full fine-tuning can be more effective it could be sample inefficient as was seen in Table 2. Further, it can be noted that the FID score remains closer to the baseline for intermediate step fine-tuning since the pre-trained model itself is used for final few denoising steps.

D.2. Fine-Tuning with ReDeNoise

ReDeNoise is an inference-time refinement strategy for the IGD framework proposed in (Anil et al., 2025). The key idea is to further noise and denoise for a set number of timesteps *after generating samples* - this was theoretically demonstrated to correct for errors in the later denoising steps. Note that ReDeNoise can be applied on top of fine-tuned models as well. We implement this and report results for 10000 generations in Table 4. It can be seen that again fine-tuning for $3T/4$ timesteps is the most beneficial. Further, note that applying ReDeNoise for the last $T/4$ timesteps *further adds improvements* for $3T/4$ step fine-tuning since the fine-tuned model is used only for the first $3T/4$ denoising steps. Hence, ReDeNoise is *complementary* to intermediate time step rejection sampling fine-tuning.

Model	Mol: Stability	Uniqueness
Baseline	89.21	96.05
$3T/4$ steps FT	**92.15**	95.46
T steps FT	90.78	96.84

Table 4. Additional Fine-tuning (FT) results for Molecule Generation