EECS 182 Deep Neural Networks
Fall 2025 Anant Sahai and Gireeja Ranade

Homework 13

This homework is due on Wednesday December 10th, at 10:59PM.

1. DDPM/DDIM Fun: From a Gaussian

Consider generative diffusion models under the simplifying assumption that the target distribution we wish to sample from is a Gaussian with zero mean and small variance σ^2, where $\sigma^2 \ll 1$. The forward process is defined to run from time $t = 0$ to $t = 1$.

Suppose the forward diffusion process starts at $X_0 \sim \mathcal{N}(0, \sigma^2)$ and at each small interval of length Δt adds independent Gaussian noise with mean zero and variance $\Delta t \ll \sigma^2$.

(a) **What is the distribution of X_1?**
(HINT: Remember, you can think of $\Delta t = \frac{1}{T}$ for some large T and X_1 as the final result after doing T steps of adding independent Gaussian noise to the initial realization of X_0.)

Solution: The total noise added over $[0, 1]$ has variance 1, so:

$$x_1 \sim \mathcal{N}(0, \sigma^2 + 1) \approx \mathcal{N}(0, 1) \text{ since } \sigma^2 \ll 1.$$

This is because we have T different independent Gaussians of noise being added together each of variance $\Delta t = \frac{1}{T}$.

(b) Looking at the forward diffusion process around time t, **what are the marginal distributions of $X_{t-\Delta t}$ and X_t?**

Solution: Marginals:

$$X_t \sim \mathcal{N}(0, \sigma^2 + t)$$
$$X_{t-\Delta t} \sim \mathcal{N}(0, \sigma^2 + t - \Delta t)$$

These are obtainable immediately from the same argument as the previous part. Except instead of 1, we have t. We can think if this as tT steps of adding independent Gaussian noise of variance $\Delta t = \frac{1}{T}$. Doing one step less is just Δt less variance.

(c) In the previous part, the conditional distribution of $(X_{t-\Delta t} | X_t = x_t) \sim \mathcal{N}\left(\frac{\sigma^2+t-\Delta t}{\sigma^2+t} x_t, \frac{(\sigma^2+t-\Delta t)\Delta t}{\sigma^2+t}\right)$.
Simplify and approximate the variance $\frac{(\sigma^2+t-\Delta t)\Delta t}{\sigma^2+t}$ of this conditional distribution when $\Delta t \ll \sigma^2 \ll 1$.

Solution:

$$\text{Var}(x_{t-\Delta t} \mid x_t) = \frac{(\sigma^2 + t - \Delta t)\Delta t}{\sigma^2 + t} = \Delta t(1 - \frac{\Delta t}{\sigma^2 + t}) \approx \Delta t$$

(d) This part asks you to see what happens if you try to do reverse denoising naively without applying stochastic noise at each reverse diffusion step. Suppose we start with a sample $X_1 \sim \mathcal{N}(0, 1)$ and iteratively apply only the conditional mean mapping backward to time $t = 0$ to get a sample of $\widehat{X}_0$.

What is the resulting distribution of $\widehat{X}_0$? Remember that $\Delta t \ll \sigma^2 \ll 1$ if you want to make any approximations or to interpret the result. Show your work.

(HINT 1: Consider $T = \frac{1}{\Delta t}$ and construct the appropriate product to see how the final $\widehat{X}_0$ is distributed. Do you notice any telescoping?)

(HINT 2: You can use the information given in the next part to check your work here.)

Solution: Think of this as T steps. Starting at $k = T-1$ and working down to $k = 0$. The factor applied at step k is the one corresponding to $t = (k+1)\Delta t$ and is:

$$\alpha_k = \frac{\sigma^2 + k\Delta t}{\sigma^2 + (k+1)\Delta t}$$

Taking a product of these

$$\begin{aligned}\prod_{k=0}^{T-1} \alpha_k &= \left(\frac{\sigma^2}{\sigma^2+\Delta t}\right)\left(\frac{\sigma^2+\Delta t}{\sigma^2+2\Delta t}\right)\left(\frac{\sigma^2+2\Delta t}{\sigma^2+3\Delta t}\right)\cdots\left(\frac{\sigma^2+(T-2)\Delta t}{\sigma^2+(T-1)\Delta t}\right)\left(\frac{\sigma^2+(T-1)\Delta t}{\sigma^2+T\Delta t}\right) \\ &= \frac{\sigma^2}{\sigma^2+T\Delta t} = \frac{\sigma^2}{\sigma^2+1}\end{aligned}$$

Where the denominator of one term cancels the numerator of the subsequent term — leaving only the first numerator and the last denominator.

Combining everything,

$$\widehat{X}_0 = \left(\frac{\sigma^2}{\sigma^2+1}\right) X_1 \Rightarrow \widehat{X}_0 \sim \mathcal{N}\left(0, \left(\frac{\sigma^2}{\sigma^2+1}\right)^2\right) \approx \mathcal{N}(0, \sigma^4)$$

Since $\sigma^2 \ll 1$, the square of it is tiny and this is approximately always equal to zero. The variance is far too small.

(e) In the previous part, you should have found that $\widehat{X}_0$ has far too little variance if you don't add independent stochastic noise along the way in DDPM-style reverse diffusion.

Let $T = \frac{1}{\Delta t}$. Suppose now that we add independent $\mathcal{N}(0, \Delta t)$ noise at each of T reverse diffusion steps. It turns out that

$$\text{Var}(\widehat{X}_0) = \left(\frac{\sigma^2}{\sigma^2+1}\right)^2 + \Delta t \sum_{k=0}^{T-1} \left(\frac{\sigma^2}{\sigma^2+k\Delta t}\right)^2. \tag{1}$$

Take the limit $\Delta t \to 0$ and **write Var$(\widehat{X}_0)$ involving an integral assuming** $0 < \Delta t \ll \sigma^2 \ll 1$**.** And then **evaluate it approximately.**

(HINT 1: For evaluating the integral, $\int \frac{1}{(1+t)^2} dt = C - \frac{1}{1+t}$.)

(HINT 2: You know that the answer should come out to be close to our desired variance of σ^2 and you can use that to check your work.)

Solution: Treating Δt as dt and viewing the second term in (1) as a Riemann sum:

$$\text{Var}(x_0) \approx \left(\frac{\sigma^2}{\sigma^2+1}\right)^2 + \int_0^1 \left(\frac{\sigma^2}{\sigma^2+t}\right)^2 dt$$

$$= \sigma^4 \left(\frac{1}{\sigma^2+1}\right)^2 + \sigma^4 \int_0^1 \left(\frac{1}{\sigma^2+t}\right)^2 dt$$
$$= \sigma^4 \left(\frac{1}{\sigma^2+1}\right)^2 + \sigma^4 (\frac{1}{\sigma^2} - \frac{1}{\sigma^2+1})$$
$$= \sigma^2 + \sigma^4 \left(\frac{1}{(\sigma^2+1)^2} - \frac{1}{\sigma^2+1}\right) = \sigma^2 - \sigma^4 \left(\frac{\sigma^2}{(\sigma^2+1)^2}\right) \approx \sigma^2$$

Where the final approximation comes from the fact that $\sigma^6 \ll \sigma^2$.

The interesting thing is that we approximately recover the desired variance of σ^2.

(f) Notice that the conditional mean step in DDPM is approximately a $\Delta t/t$ step toward the ideal predictor $\widehat{X}_0^* = \frac{\sigma^2}{\sigma^2+t} X_t$. Namely that

$$\widehat{X}_{t-\Delta t}^{\text{DDPM}} = \widehat{x}_t + \frac{\Delta t}{t}(\widehat{X}_0^* - \widehat{x}_t) + \text{noise} = \left(1 - \frac{\Delta t}{\sigma^2+t}\right)\widehat{x}_t + \text{noise}$$

This can be incrementally viewed as a deterministic DDPM step of $-\frac{\Delta t}{\sigma^2+t} x_t$ together with a noise step.

DDIM takes no noise step, but a smaller time-varying deterministic step:

$$x_{t-\Delta t}^{\text{DDIM}} = x_t + \eta(t, \Delta t)(\text{deterministic DDPM step}) \quad \text{with } \eta(t, \Delta t) = \frac{\sqrt{t}}{\sqrt{t-\Delta t} + \sqrt{t}}$$

Describe briefly, if you were simply given a dataset of samples $s_1, s_2, \ldots, s_n$ drawn from the same distribution as our desired X_0 distribution, how you would train a neural network to estimate the function $g(x_t, t)$ that in this case, turns out to be $\frac{\sigma^2}{\sigma^2+t} x_t$ in analytic form. **Specifically, what are the inputs to the neural net, how do you generate a batch of them, and how would you compute a loss on the outputs of the neural net** for running an optimization algorithm like AdamW to set the parameters of the neural net?

Solution: The inputs to the neural net are the position $t \in [0, 1]$ and the x_t.

To generate a batch of size k, draw k iid copies for t_i from a uniform random variable on the interval $[0, 1]$ along with Gaussian noise z_i from $\mathcal{N}(0, t_i)$. Also draw k random training points s_{ℓ_i} by drawing the ℓ_i iid from the discrete uniform distribution on $\{1, 2, \ldots, n\}$. The $(t_i, s_{\ell_i} + z_i)$ define the input pairs.

To compute the loss for the neural network $f_\theta(t, x)$, use the squared loss $(s_{\ell_i} - f_\theta(t_i, s_{\ell_i} + z_i))^2$ — i.e. try to predict the original sample from the time and noisy version.

Note: scalings used to make these inputs have more bounded ranges are also acceptable and practically useful.

This approach does not depend on the actual distribution for X_0 that we want to learn — all that dependence is through the training set.

(g) Assuming you had exact analytic form access to $g(x_t, t) = \frac{\sigma^2}{\sigma^2+t} x_t$ in the previous part, **approximate the DDIM step for $\Delta t \ll t$, with no assumption about σ^2 vs t.**

(HINT: Use the fact that $\Delta t \ll t$ to approximate $\eta(t, \Delta t)$ as an appropriate constant.)

Solution:

$$x_{t-\Delta t}^{\text{DDIM}} = \left(1 - \frac{\eta(t, \Delta t)\Delta t}{\sigma^2+t}\right) x_t$$

Using expansion:

$$\eta(t,\Delta t) \approx \frac{1}{2} + \frac{\Delta t}{8t} \Rightarrow x_{t-\Delta t}^{\text{DDIM}} \approx x_t \left(1 - \frac{\Delta t}{2(\sigma^2+t)} - \frac{\Delta t^2}{8t(\sigma^2+t)}\right)$$

This is basically just $x_t \left(1 - \frac{\Delta t}{2(\sigma^2+t)}\right)$.

(h) Compute $\widehat{X}_0$ from the random sample $\widehat{X}_1$ (drawn from $\mathcal{N}(0,1)$) by applying all $T = \frac{1}{\Delta t}$ DDIM steps from the previous part and expressing the total contraction as a product. Then turn it into a sum via logarithms and take the $\Delta t \to 0$ limit, using integrals as needed to evaluate the resulting distribution for $\widehat{X}_0$. **Show your work.**

(HINT 1: You know that the answer is supposed to be approximately $\mathcal{N}\left(0,\sigma^2\right)$ under our assumption that $\sigma^2 \ll 1$. Use this to check your work.)

(HINT 2: Remember $\ln(1-x) \approx -x$ if $0 < x \ll 1$.)

(HINT 3: The log of a limit is the limit of logs.)

Solution:

$$\begin{aligned}\widehat{X}_0 &= X_1 \cdot \prod_{k=1}^{T} \left(1 - \frac{\eta(k\Delta t, \Delta t)\Delta t}{\sigma^2 + k\Delta t}\right) \\ &\approx X_1 \cdot \prod_{k=1}^{T} \left(1 - \frac{\Delta t}{2(\sigma^2 + k\Delta t)}\right)\end{aligned}$$

Because it is a scaling of a Gaussian random variable, we know that $\widehat{X}_0$ is a Gaussian random variable with zero mean. It is defined by its variance which is $(\prod_{k=1}^{T}\left(1 - \frac{\Delta t}{2(\sigma^2+k\Delta t)}\right))^2$.

Taking logs and limits of the product part — noticing that every term is strictly positive since $k\Delta t = k\Delta t$ and $\Delta t \ll \sigma_2$ which allows one to use the approximation $\ln(1-x) \approx -x$, gives us:

$$\begin{aligned}\ln\left(\prod_{k=1}^{T}\left(1 - \frac{\Delta t}{2(\sigma^2+k\Delta t)}\right)\right)^2 &= 2\sum_{k=1}^{T} \ln\left(1 - \frac{\Delta t}{2(\sigma^2+k\Delta t)}\right) \\ &\approx 2\sum_{k=1}^{T} -\frac{\Delta t}{2(\sigma^2+k\Delta t)} \\ &\approx -2\int_0^1 \frac{1}{2(\sigma^2+t)}dt \\ &= -\left(\ln\left(\sigma^2+1\right) - \ln\sigma^2\right) \\ &= \ln\frac{\sigma^2}{\sigma^2+1}\end{aligned}$$

$$\Rightarrow X_0 = X_1 \cdot \left(\frac{\sigma^2}{\sigma^2+1}\right)^{1/2} \Rightarrow X_0 \sim \mathcal{N}\left(0, \frac{\sigma^2}{\sigma^2+1}\right) \approx \mathcal{N}\left(0,\sigma^2\right)$$

as desired.

2. Honey, Where's My Reward Model?

In the standard Reinforcement Learning from Human Feedback (RLHF) pipeline like in Ziegler et al, there are usually three phases:

1 **SFT:** We perform Supervised Fine-Tuning (SFT) on a pre-trained LM using a dataset of high-quality demonstrations (e.g., summarization, sentiment analysis, etc.) to obtain a model π^{SFT}.

2 **Reward Modeling:** Next, we use the SFT model to generate pairs of answers $(y_1, y_2) \sim \pi^{\text{SFT}}(y \mid x)$ for various prompts x. Human labelers rank these pairs, producing a dataset of preferences $\mathcal{D} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^N$, where y_w denotes the preferred (winning) output and y_l the dispreferred (losing) output amongst (y_1, y_2), respectively. Crucially, we assume these preferences are generated by some latent (hidden) reward model $r^*(x, y)$, which we do not have access to. Instead, we parameterize a proxy reward model $r_\phi(x, y)$ to approximate it. However, we face a challenge: How do we model these preferences, which are inherently discrete ($y_w \succ y_l$), using a reward model that outputs continuous scalars?

   To bridge this gap, we employ the Bradley-Terry (BT) model. The BT model assumes that the probability of a human preferring one response over another is given by the sigmoid of the difference in their latent rewards:

$$p^*(y_w \succ y_l|x) = \sigma(r^*(x, y_w) - r^*(x, y_l)) = \frac{\exp\big(r^*(x, y_w)\big)}{\exp\big(r^*(x, y_w)\big) + \exp\big(r^*(x, y_l)\big)}$$

   This formulation allows us to frame preference learning as a binary classification problem because it treats the reward difference as a logit: if $r^*(x, y_w)$ is significantly higher than $r^*(x, y_l)$, the sigmoid outputs a probability near 1, matching the "label" that y_w is the winner. As a result, we can estimate the parameters ϕ via Maximum Likelihood Estimation (MLE) by minimizing the negative log-likelihood loss:

$$\mathcal{L}_R(\phi) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right]$$

   In the context of LMs, $r^*(x, y)$ is often initialized from the SFT model $\pi^{\text{SFT}}(y \mid x)$ with the addition of a classification head to produce a single scalar prediction for the reward value. Furthermore, to ensure a reward function with lower variance, prior works normalize the rewards, such that $\mathbb{E}_{(x,y)\sim\mathcal{D}}[r_\phi(x, y)] = 0$ for all x.

3 **RL Fine-Tuning:** Finally, we use learned reward function as feedback to the language model. The optimization problem is formulated as

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)}\left[r_\phi(x, y)\right] - \beta\mathbb{D}_{\text{KL}}[\pi_\theta(y \mid x)\|\pi_{\text{ref}}(y \mid x)]$$

   where β is a parameter controlling the deviatation from the base reference policy π_{ref}, namely the initial SFT model π^{SFT}. In practice, the language model policy π_θ is also initialized to π^{SFT}. Due to the discrete nature of language generation, the objective is not differentiable so it is typically optimized using reinforcement learning. The standard approach has been to construct the reward function $r(x, y) = r_\phi(x, y) - \beta(\log \pi_\theta(y \mid x) - \log \pi_{\text{ref}}(y \mid x))$, and maximize using PPO.

Unfortunately, this three-step process is brittle and computationally expensive. This where Direct Preference Optimization (DPO) comes in.

In this problem, you will derive **Direct Preference Optimization (DPO)**, a method introduced by Rafailov et al. that provides a simple yet effective approach for policy optimization using preferences directly.

(a) As a warm up, find the optimal p^* for the optimization problem:

$$\min_{p \in \mathcal{P}} \mathbb{D}_{\mathrm{KL}}(p \| q)$$

where

$$\mathbb{D}_{\mathrm{KL}}(p \| q) = \mathbb{E}_{x \sim p(x)} \log \left(\frac{p(x)}{q(x)} \right) = \sum_x p(x) \log \left(\frac{p(x)}{q(x)} \right) \geq 0$$

is the KL divergence of p from q.
Hint: $\mathbb{D}_{\mathrm{KL}}(p \| q) = 0$ *iff* $p = q$

Solution: Since the KL divergence is non-negative ($\mathbb{D}_{\mathrm{KL}}(p \| q) \geq 0$) and equals zero if and only if $p(x) = q(x)$ for all x, $p^* = q$.

(b) As discussed earlier, we wish to find a language model policy π_θ that maximizes the expected reward while preventing the model from drifting too far from the reference model:

$$\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} \left[r_\phi(x, y) \right] - \beta \mathbb{D}_{\mathrm{KL}}[\pi_\theta(y \mid x) \| \pi_{\mathrm{ref}}(y \mid x)]$$

Show that the optimal solution $\pi_{\theta^*}(y \mid x)$ takes the form:

$$\pi_{\theta^*}(y|x) = \frac{1}{Z(x)} \pi_{\mathrm{ref}}(y \mid x) \exp \left(\frac{1}{\beta} r_\phi(x, y) \right)$$

where $Z(x) = \sum_y \pi_{\mathrm{ref}}(y \mid x) \exp \left(\frac{1}{\beta} r_\phi(x, y) \right)$ is the partition function.
Hint 1: Transform the problem from maximization to minimization by negating the objective and dividing by β. Recall that the optimal solution π_{θ^*} is invariant to scaling or shifting the objective function.
Hint 2: Expand the definition of $\mathbb{D}_{\mathrm{KL}}$ and combine it with the reward term so that the entire objective is a single expectation over $y \sim \pi_\theta(y \mid x)$.
Hint 3: Define the distribution $\pi^*(y \mid x) = \frac{1}{Z(x)} \pi_{\mathrm{ref}}(y \mid x) \exp \left(\frac{1}{\beta} r_\phi(x, y) \right)$. Multiply and divide your objective term by $Z(x)$ so you can substitute π^* into the equation.
Hint 4: Finally, rearrange the objective into the form $\min_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}} \left[\mathbb{D}_{\mathrm{KL}}(\pi_\theta \| \pi^*) - C \right]$. From here, your answer from part (a) will be helpful.

Solution: We expand the objective function with respect to π_θ:

$$\begin{aligned}
&\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} \left[r_\phi(x, y) \right] - \beta \mathbb{D}_{\mathrm{KL}}[\pi_\theta(y \mid x) \| \pi_{\mathrm{ref}}(y \mid x)] \\
&\quad = \max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} \left[r_\phi(x, y) \right] - \beta \mathbb{E}_{y \sim \pi_\theta(y|x)} \log \left(\frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} \right) \\
&\quad = \max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{y \sim \pi_\theta(y|x)} \left[r_\phi(x, y) - \beta \log \left(\frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} \right) \right] \\
&\quad = \min_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{y \sim \pi_\theta(y|x)} \left[\log \left(\frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} \right) - \frac{1}{\beta} r_\phi(x, y) \right] \\
&\quad = \min_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{y \sim \pi_\theta(y|x)} \left[\log \left(\frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x) \exp \left(\frac{1}{\beta} r_\phi(x, y) \right)} \right) \right]
\end{aligned}$$

$$= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} \left[\log \left(\frac{\pi_\theta(y \mid x)}{\frac{Z(x)}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r_\phi(x,y)\right)} \right) \right]$$

$$= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} \left[\log \left(\frac{\pi_\theta(y \mid x)}{\frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r_\phi(x,y)\right)} \right) - \log\big(Z(x)\big) \right]$$

where $Z(x)$ is the partition function defined as

$$Z(x) = \sum_y \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r_\phi(x,y)\right)$$

Note that the partition function is a function that only depends on x and the reference policy π_{ref}, but does not depend on the policy π_θ. With this in mind, we now define

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r_\phi(x,y)\right)$$

which is a valid probability distribution because $\pi^*(y \mid x) \geq 0$ for all y and $\sum_y \pi^*(y \mid x) = 1$. Finally, we re-organize the objective to get

$$= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \left[\mathbb{E}_{y\sim\pi_\theta(y|x)} \left[\log \left(\frac{\pi_\theta(y \mid x)}{\pi^*(y \mid x)} \right) \right] - \log\big(Z(x)\big) \right]$$

$$= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \left[\mathbb{D}_{\text{KL}}(\pi_\theta(y \mid x) \| \pi^*(y \mid x)) - \log\big(Z(x)\big) \right]$$

Since $Z(x)$ does not depend on π_θ, we shift our attention toward minimizing the first KL term. From part (a), we know the optimal solution is $\pi_{\theta^*} = \pi^*$, and thus we are done.

Note that this optimization problem can be solved under any reward function $r(x,y)$, reference model π_{ref}, and a general non-parametric policy class

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi} \left[r(x,y) \right] - \beta \mathbb{D}_{\text{KL}}[\pi(y \mid x) \| \pi_{\text{ref}}(y \mid x)]$$

where $\pi^*(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x,y)\right)$ is the optimal policy.

(c) Even though we've obtained the optimal policy $\pi_{\theta^*}(y \mid x)$, this particular representation of it is hard to utilize in practice. Why? *Hint: How large can the output space $\mathcal{Y}$ be?*

Solution: The culprit here is the partition function $Z(x)$, which requires summing over all possible outputs $y \in \mathcal{Y}$. In practice, this set can be astronomically large, making computation of $Z(x)$ incredibly expensive. To see why, consider a model with a vocabulary $\mathcal{V}$ of size $|\mathcal{V}|$. Then the output space $\mathcal{Y}$ contains all sequences of tokens from $\mathcal{V}$, and its size grows exponentially with the maximum sequence length L that the model can generate:

$$|\mathcal{Y}| = \Theta(|\mathcal{V}|^L)$$

For example, GPT-3 has a vocabulary size of 50257 with a context window of 2048 tokens. This means that the size of the output space can be as large as $50257^{2048} \approx 1.1 \times 10^{9628}$. Wow!!!

(d) Using the expression for the optimal policy π_{θ^*} derived in part (b), express the reward function $r_\phi(x, y)$ in terms of π_{θ^*}, π_{ref}, and $Z(x)$.

Solution: We start with the expression for the optimal policy π_{θ^*}:

$$\pi_{\theta^*}(y|x) = \frac{1}{Z(x)}\pi_{\text{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r_\phi(x, y)\right)$$

then taking the logarithm of both sides, we get:

$$\log \pi_{\theta^*}(y|x) = -\log Z(x) + \log \pi_{\text{ref}}(y \mid x) + \frac{1}{\beta}r_\phi(x, y)$$

then after rearranging, we get the reward function $r_\phi(x, y)$ as:

$$\frac{1}{\beta}r_\phi(x, y) = \log\left(\frac{\pi_{\theta^*}(y \mid x)}{\pi_{\text{ref}}(y \mid x)}\right) + \log Z(x)$$
$$r_\phi(x, y) = \beta\log\left(\frac{\pi_{\theta^*}(y \mid x)}{\pi_{\text{ref}}(y \mid x)}\right) + \beta\log Z(x)$$

(e) With this reparametrization of $r_\phi(x, y)$, substitute it into the Bradley-Terry model to obtain the probability of human preference data in terms of the optimal policy. Furthermore, show that the partition function $Z(x)$ cancels out in your computation. Why is this desirable?

Solution: For the Bradley-Terry model, we have

$$p_{\theta^*}(y_1 \succ y_2|x) = \frac{\exp\big(r_\phi(x, y_1)\big)}{\exp\big(r_\phi(x, y_1)\big) + \exp\big(r_\phi(x, y_2)\big)}$$

Substituting this in, we get

$$\begin{aligned}
&p_{\theta^*}(y_1 \succ y_2|x) \\
&= \frac{\exp\left(\beta\log\left(\frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right) + \beta\log\big(Z(x)\big)\right)}{\exp\left(\beta\log\left(\frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right) + \beta\log\big(Z(x)\big)\right) + \exp\left(\beta\log\left(\frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)}\right) + \beta\log\big(Z(x)\big)\right)} \\
&= \frac{1}{1 + \exp\left(\beta\log\left(\frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)}\right) - \beta\log\left(\frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)\right)} \\
&= \sigma\left(\beta\log\left(\frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right) - \beta\log\left(\frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)}\right)\right).
\end{aligned}$$

Since the partition function $Z(x)$ cancels out, computing the probability $p_{\theta^*}(y_1 \succ y_2|x)$ is computationally feasible! As discussed in part (c), computing $Z(x)$ is almost impossible in practice.

Now that we have the probability of human preference data in terms of the optimal policy rather than the reward model, we can formulate the Maximum Likelihood objective for the parameterized policy π_θ

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

which is exactly the result we wanted!

Recall the complex RLHF pipeline we started with: **SFT** $\rightarrow$ **Reward Modeling** $\rightarrow$ **RL Fine-Tuning**. By deriving the equation above, we have bypassed the need for an explicit reward model $r_\phi(x, y)$ and the RL loop entirely. Instead, we are fitting an *implicit* reward model (defined by the log-ratio of our policy and the reference) using a simple binary cross-entropy loss.

Crucially, because we derived this objective by algebraically solving for the reward in part (c), the policy π_θ that minimizes this loss is guaranteed to be the optimal policy for that implicit reward function. We have thus reduced the RLHF pipeline to a standard supervised classification problem!

(f) To verify how this optimization works in practice, let the implicit reward be defined as $\hat{r}_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$. Derive the gradient of the DPO loss with respect to the parameters θ, i.e., $\nabla_\theta \mathcal{L}_{\text{DPO}}$. Afterwards, interpret the role of the weighting term $\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w))$ in the $\nabla_\theta \mathcal{L}_{\text{DPO}}$. Does the gradient update the model more when it is correct, or when it is incorrect?
Hint: Recall that $\sigma'(z) = \sigma(z)(1 - \sigma(z)) = \sigma(z)\sigma(-z)$.

Solution:

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

We can rewrite the RHS of the above equation as

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\frac{\sigma'(u)}{\sigma(u)} \nabla_\theta(u) \right]$$

where $u = \beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}$.
Using the properties of sigmoid function $\sigma'(x) = \sigma(x)(1 - \sigma(x))$ and $\sigma(-x) = 1 - \sigma(x)$, we obtain the final gradient

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

After using the reward substitution of $\hat{r}_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$ we obtain the final form of the gradient as

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}})$$
$$= -\beta \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\underbrace{\sigma\left(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w)\right)}_{\text{higher weight when reward estimate is wrong}} \left[\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increases the likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi(y_l \mid x)}_{\text{decreases the likelihood of } y_l} \right] \right]$$

Interpretation: The term $\nabla_\theta \log \pi_\theta(y_w \mid x) - \nabla_\theta \log \pi_\theta(y_l \mid x)$ is the update direction: it pushes the model to increase the likelihood of the winner y_w and decrease the loser y_l. Furthermore, the scalar term $\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w))$ acts as a dynamic weight:

- If the model is **correct** (gives y_w much higher reward than y_l), then $\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w) \rightarrow -\infty$ and $\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w)) \rightarrow 0$. The gradient vanishes, meaning the model stops learning from examples it has already "solved."

- If the model is **incorrect** (gives y_l higher reward than y_w), then $\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w) > 0$ and $\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w)) \to 1$ the worse that kind of error is. The gradient is strong, forcing the model to correct its mistake.

(g) Throughout this problem, we have worked with the Bradley-Terry model to model human preferences. However, in many real-world scenarios, labelers rank multiple model outputs $y_1, \ldots, y_K$ rather than just comparing pairs. This is modeled by the Plackett-Luce model, which is a generalization of the Bradley-Terry model. In a similar fashion, the PL model stipulates that when presented with a set of possible choices, people prefer a choice with a probability prportional to the value of some latent reward function for that choice. In our context, when presented with a prompt x and a set of K answers $y_1, \ldots, y_K \sim \pi^{\text{SFT}}(y \mid x)$, a user would output a permutation $\tau \colon \{1, \ldots, K\} \to \{1, \ldots, K\}$ indicating the preferred order of the choices. The probability of observing this ranking is given by:

$$p_{\theta^*}(\tau | y_1, \ldots, y_K, x) = \prod_{k=1}^{K} \frac{\exp\Big(r_\phi(x, y_{\tau(k)})\Big)}{\sum_{j=k}^{K} \exp\Big(r_\phi(x, y_{\tau(j)})\Big)}$$

Using the implicit reward parameterization derived in part (d) ($r_\phi(x, y) = \beta \log \frac{\pi_{\theta^*}(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x)$), show that the partition function $Z(x)$ cancels out here as well. Write the final Plackett-Luce probability solely in terms of π_{θ^*} and π_{ref}.

Once you've written out that final probability, assuming we have access to a dataset $\mathcal{D} = \{(\tau^{(i)}, x^{(i)}, \{y_j^i\}_{j=1}^K)\}_{i=1}^N$ of prompts and user-specified rankings, we can use a parameterized model and optimize this objective with maximum-likelihood:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(\tau, y_1, \ldots, y_K, x\tau) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_{\tau(k)} \mid x)}{\pi_{\text{ref}}(y_{\tau(k)} \mid x)} - \beta \log \frac{\pi_\theta(y_{\tau(j)}|x)}{\pi_{\text{ref}}(y_{\tau(j)}|x)} \right) \right]$$

Solution: First, let us simplify the exponential of the reward function using the reparameterization given in the problem statement:

$$\begin{aligned}
\exp\big(r_\phi(x, y)\big) &= \exp\left(\beta \log \frac{\pi_{\theta^*}(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x) \right) \\
&= \exp\left(\beta \log \frac{\pi_{\theta^*}(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right) \cdot \exp\big(\beta \log Z(x)\big) \\
&= \exp\left(\beta \log \frac{\pi_{\theta^*}(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right) \cdot Z(x)^\beta
\end{aligned}$$

Now, consider the term for the k-th ranked item in the Plackett-Luce product. Substituting the expression derived above:

$$\frac{\exp\Big(r_\phi(x, y_{\tau(k)})\Big)}{\sum_{j=k}^{K} \exp\Big(r_\phi(x, y_{\tau(j)})\Big)} = \frac{Z(x)^\beta \cdot \exp\left(\beta \log \frac{\pi_{\theta^*}(y_{\tau(k)}|x)}{\pi_{\text{ref}}(y_{\tau(k)}|x)} \right)}{\sum_{j=k}^{K} \left[Z(x)^\beta \cdot \exp\left(\beta \log \frac{\pi_{\theta^*}(y_{\tau(j)}|x)}{\pi_{\text{ref}}(y_{\tau(j)}|x)} \right) \right]}$$

$$= \frac{Z(x)^\beta \cdot \exp\left(\beta \log \frac{\pi_{\theta^*}(y_{\tau(k)}|x)}{\pi_{\text{ref}}(y_{\tau(k)}|x)}\right)}{Z(x)^\beta \cdot \sum_{j=k}^{K} \exp\left(\beta \log \frac{\pi_{\theta^*}(y_{\tau(j)}|x)}{\pi_{\text{ref}}(y_{\tau(j)}|x)}\right)}$$

The term $Z(x)^\beta$ appears in both the numerator and the denominator (factored out of the sum), so it cancels out completely.

Thus, the final probability depends only on the optimal policy π_{θ^*} and the reference policy π_{ref}:

$$p_{\theta^*}(\tau|y_1, \ldots, y_K, x) = \prod_{k=1}^{K} \frac{\exp\left(\beta \log \frac{\pi_{\theta^*}(y_{\tau(k)}|x)}{\pi_{\text{ref}}(y_{\tau(k)}|x)}\right)}{\sum_{j=k}^{K} \exp\left(\beta \log \frac{\pi_{\theta^*}(y_{\tau(j)}|x)}{\pi_{\text{ref}}(y_{\tau(j)}|x)}\right)}$$

3. Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student!
We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

(a) **What sources (if any) did you use as you worked through the homework?**

(b) **If you worked with someone on this homework, who did you work with?**
List names and student ID's. (In case of homework party, you can also just describe the group.)

(c) **Roughly how many total hours did you work on this homework?**

The coverage of the following question is redundant. It is presented here to aid with exam preparation.

4. Diffusion Models

Previously in discussion, we considered sampling from a discrete distribution. Let's now see how iteratively adding Gaussian noise to a data point leads to a noisy sequence, and how the reverse process refines noise to generate realistic samples.

The classes of generative models we've considered so far (VAEs, GANs), typically introduce some sort of bottleneck (*latent representation* $\mathbf{z}$) that captures the essence of the high-dimensional sample space ($\mathbf{x}$). An alternate view of representing probability distributions $p(\mathbf{x})$ is by reasoning about the *score function* i.e. the gradient of the log probability density function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$.

Given a data point sampled from a real data distribution $\mathbf{x}_0 \sim q(\mathbf{x})$, let us define a *forward diffusion process* iteratively adding small amount of Gaussian noise to the sample in T steps, producing a sequence of noisy samples $\mathbf{x}_1, ..\mathbf{x}_T$.

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t I) \qquad q(\mathbf{x}_{1:T}|x_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{2}$$

The data sample $\mathbf{x}_0$ gradually loses its distinguishable features as the step t becomes larger. Eventually when $T \to \infty$, $\mathbf{x}_T$ is equivalent to an isotropic Gaussian distribution. (You can assume $\mathbf{x}_0$ is Gaussian).

To generative model is therefore the *reverse diffusion process*, where we sample noise from an isotropic Gaussian, and iteratively refine it towards a realistic sample by reasoning about $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$.

(a) **Anytime Sampling from Intermediate Distributions**

Given $\mathbf{x}_0$ and the stochastic process in eq. (2), **show that there exists a closed form distribution for sampling directly at the t^{th} time-step of the form**

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_0, (1-\alpha_t)I)$$

Solution: Recall the reparameterization trick, where to sample from a Gaussian $\mathbf{x} \sim \mathcal{N}(\mu, \sigma^2)$, we could consider the following sampling process:

$$\mathbf{x} = \mu + \sigma\epsilon \qquad \text{where } \epsilon \sim \mathcal{N}(0,1)$$

Therefore, defining $\gamma_t = 1 - \beta_t$, we have

$$\begin{aligned} \mathbf{x}_t &= \sqrt{\gamma_t}\mathbf{x}_{t-1} + \sqrt{(1-\gamma_t)}\epsilon_{t-1} && \text{where } \epsilon_{t-1} \sim \mathcal{N}(0, I) \\ &= \sqrt{\gamma_t}\Big(\sqrt{\gamma_{t-1}}\mathbf{x}_{t-2} + \sqrt{(1-\gamma_{t-1})}\epsilon_{t-2}\Big) + \sqrt{(1-\gamma_t)}\epsilon_{t-1} && \text{where } \epsilon_{t-2} \sim \mathcal{N}(0, I) \end{aligned}$$

To simplify this, recall the following lemma, where mixing two Gausssians $\mathcal{N}(0, \sigma_1^2)$ and $\mathcal{N}(0, \sigma_2^2)$ gives a Gaussian $\mathcal{N}(0, \sigma_1^2 + \sigma_2^2)$. Therefore, mixing samples ϵ_1, ϵ_2. Building on this insight, we can combine the noise components ϵ_1, ϵ_2 into a new random variable:

$$\begin{aligned} \hat{\epsilon}_{t-2} &\sim \mathcal{N}(0, (\gamma_t(1-\gamma_{t-1}) + (1-\gamma_t))I) \\ &\sim \mathcal{N}(0, (1-\gamma_t\gamma_{t-1})I) \\ \therefore \mathbf{x}_t &= \sqrt{\gamma_t\gamma_{t-1}}\mathbf{x}_{t-2} + \sqrt{(1-\gamma_t\gamma_{t-1})}\hat{\epsilon}_{t-2} \end{aligned}$$

Unrolling this recursion, we would get the base case, where for $\mathbf{x}_0$ the samples are

$$\mathbf{x}_t = \sqrt{\Pi_{i=1}^t \gamma_i}\mathbf{x}_0 + \sqrt{1 - \Pi_{i=1}^t \gamma_i}\epsilon$$

Therefore, by introducting $\alpha_t = \Pi_{i=1}^t \gamma_i$ we get that

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_0, (1-\alpha_t)I)$$

(b) **Reversing the Diffusion Process**

Reversing the diffusion process from *real* to *noise* would allow us to sample from the real data distribution. In particular, we would want to draw samples from $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$. **Show that given $\mathbf{x}_0$, the reverse conditional probability distribution is tractable and given by**

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \mu(\mathbf{x}_t, \mathbf{x}_0), \hat{\beta}_t I)$$

(Hint: Use Bayes Rule on eq. (2) , assuming that $\mathbf{x}_0$ is drawn from Gaussian $q(\mathbf{x})$)

Solution: Simple proof by the reparametrization trick on Wikipedia.

Solution: Applying Bayes rule on $q(x_t|x_{t-1}, x_0)$ we get the following expression

$$q(x_{t-1}|x_t, x_0) = q(x_t|x_{t-1}, x_0)\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}$$

From part (a) we know the densities as

$$q(x_t|x_0) \sim \mathcal{N}(\sqrt{\alpha_t}x_0, (1-\alpha_t)I)$$
$$q(x_t|x_{t-1}, x_0) \sim \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

Therefore by plugging into the Bayes rule, we recover (upto proportionality constants)

$$\begin{aligned} q(x_{t-1}|x_t, x_0) &\propto \exp\Big(-\frac{1}{2}\Big\{\frac{(x_t - \sqrt{1-\beta_t}x_{t-1})^2}{\beta_t} + \frac{(x_{t-1} - \sqrt{\alpha_{t-1}}x_0)^2}{1-\alpha_{t-1}} - \frac{(x_t - \sqrt{\alpha_t}x_0)^2}{1-\alpha_t}\Big\}\Big) \\ &\propto \exp\Big(-\frac{1}{2}\Big\{\frac{x_t^2 - 2\sqrt{1-\beta_t}x_{t-1}x_t + (1-\beta_t)x_{t-1}^2}{\beta_t} + \\ &\qquad \frac{x_{t-1}^2 - 2\sqrt{\alpha_{t-1}}x_0x_{t-1} + \alpha_{t-1}x_0^2}{1-\alpha_{t-1}} - \frac{(x_t - \sqrt{\alpha_t}x_0)^2}{1-\alpha_t}\Big\}\Big) \end{aligned}$$

Simplifying the expression we get

$$q(x_{t-1}|x_t, x_0) \propto \exp\Big(-\frac{1}{2}\Big\{(\frac{1-\beta_t}{\beta_t} + \frac{1}{1-\alpha_{t-1}})x_{t-1}^2 - (\frac{2\sqrt{1-\beta_t}}{\beta_t}x_t + \frac{2\sqrt{\alpha_{t-1}}}{1-\alpha_{t-1}}x_0)x_{t-1} + H(x_t, x_0)\Big\}\Big)$$

where $H(x_t, x_0)$ is independent of x_{t-1} and therefore would be normalized out. Comparing to the expression for Gaussian $\mathcal{N}(\mu, \sigma^2)$

$$\mathcal{N}(\mu, \sigma^2) \propto \exp\Big(-\frac{1}{2}\Big\{\frac{x^2 - 2\mu x + \mu^2}{\sigma^2}\Big\}\Big)$$

we recover the expression for mean, variance of $q(x_{t-1}|x_t, x_0)$ as

$$\begin{aligned}
\hat{\beta}_t &= 1/\Big(\frac{1-\beta_t}{\beta_t} + \frac{1}{1-\alpha_{t-1}}\Big) \\
&= \frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t \qquad\qquad \Big(\text{recall } \alpha_t = \prod_{i=1}^{T}(1-\beta_t)\Big) \\
\mu(x_t, x_0) &= \Big(\frac{\sqrt{1-\beta_t}}{\beta_t}x_t + \frac{\sqrt{\alpha_{t-1}}}{1-\alpha_{t-1}}x_0\Big)\Big/\Big(\frac{1-\beta_t}{\beta_t} + \frac{1}{1-\alpha_{t-1}}\Big) \\
&= \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})}{1-\alpha_t}x_t + \frac{\beta_t\sqrt{\alpha_{t-1}}}{1-\alpha_t}x_0
\end{aligned}$$

Therefore, under our assumptions, the distribution of $q(x_{t-1}|x_t, x_0) \sim \mathcal{N}(\mu(x_t, x_0), \hat{\beta}_t I)$. □

Contributors:

- Anant Sahai.
- Patrick Mendoza.
- Kumar Krishna Agrawal.

