Diffusion model: a basic intro.

Ref:

Denoising diffusion probablistic model, Ho et al. (2020) NeurIPS.

Score-based generative modeling through stochastic differential equation, Song et al. (2021) ICLR

Generative modeling by estimating gradients of data distribution, Song and Ermon (2019) NeurIPS.

Understanding diffusion model: A unified perspective, Luo (2020)

Diffusion model: A comprehensive Survey of methods and applications, Song et al. (2023).

What are diffusion models? Lil'Log by Lilian Weng.

Tutorial on diffusion models for imaging and vision. Stanley Chan (2024).

Step-by-step diffusion: an elementary tutorial. Nakkiran et al. (2024)

1. DDPM

We start from the introduction of a class of autoencoder algorithm. We will see that the DDPM is nothing but a special VAE algorithm.

1.1. Evidence lower bound.

Like traditional autoencoder algorithm, given observed data x, we imagine there is some latent variable z as a lower-dimensional representation. Our goal is to learn a model to maximize the likelihood $p_\theta(x)$. However, optimizing $p_\theta(x)$ can be hard for some complicated distribution, thus for fixed θ, we consider an Evidence Lower bound (ELBO):

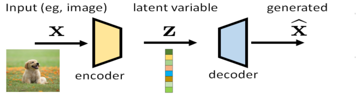
$$\log p_\theta(x) = \log \int p_\theta(x, z)\, dz$$

learnable encoder

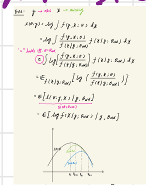
$$= \log \int \frac{p_\theta(x,z)\, q_\phi(z|x)}{q_\phi(z|x)}\, dz \quad \cdots\cdots \text{(ELBO 1)}$$

(Jensen)

$$\geq \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right)\right] \quad \cdots\cdots \text{(ELBO)}$$



Remark: From (ELBO 1) we can see a close relationship between ELBO and EM-algorithm. In EM-algorithm we use $p(z|x;\theta_{old})$ rather than the encoder $q_\phi(z|x)$.



But we are not satisfied with this derivation, because it only implies that (ELBO) is a lower bound of the log-likelihood but the Jensen's inequality mysteriously hides the reason. Let's perform another derivation:

$$
\begin{aligned}
\log p_\theta(x) &= \int q_\phi(z|x) \cdot \log p_\theta(x)\, dz \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x)\right] \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(X,Z)}{p_\theta(Z|X)}\right] \\
&= \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(X,Z)\, q_\phi(Z|X)}{p_\theta(Z|X)\, q_\phi(Z|X)}\right) \\
&= \underbrace{\mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)}_{\text{ELBO}} + \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right)
\end{aligned}
$$

$$= \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right) + \underbrace{KL\left(q_\phi(z|x) \,\|\, p_\theta(z|x)\right)}_{\geq 0}$$

$$\geq \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)$$

This derivation reveals the reason we can maximize the ELBO loss:

$$\underbrace{\log p_\theta(x)}_{\text{Likelihood: Independent with } \phi} = \overbrace{\mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)}^{\text{ELBO}} + KL\big(\underbrace{q_\phi(z|x)}_{\text{Encoder}} \,\|\, \underbrace{p_\theta(z|x)}_{\text{True posterior}}\big) \quad \cdots\cdots (RM)$$

$\text{ELBO}\uparrow \Rightarrow KL(q_\phi \| p_\theta)\downarrow$

Remark: For a fixed θ, the log-likelihood is a constant of ϕ, thus maximizing the ELBO w.r.t. ϕ will effectively push the KL-divergence between $q_\phi(z|x)$ and $p_\theta(z|x)$ to zero which is desirable.

Therefore, we consider a new objective function:

$$\boxed{\mathcal{L}(\theta) = \max_\phi(\text{ELBO}) = \max_\phi \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(X,Z)}{q_\phi(Z|X)}\right)}$$

1.2. Variational Autoencoder (VAE)

With ELBO, the default formulation of VAE is simply maximize the ELBO:

$$\max_{\theta} L(\theta) = \max_{\theta,\phi} \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right)$$

$$= \max_{\theta,\phi} \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p_\theta(x|z)\,p(z)}{q_\phi(z|x)}\right)$$

Reformulation of ELBO:

$$= \max_{\theta,\phi}\left\{\mathbb{E}_{q_\phi(z|x)}\left(\log p_\theta(x|z)\right) + \mathbb{E}_{q_\phi(z|x)}\left(\log \frac{p(z)}{q_\phi(z|x)}\right)\right\}$$

VAE:

$$\boxed{= \max_{\theta,\phi}\left\{\underbrace{\mathbb{E}_{q_\phi(z|x)}\left(\log v_\theta(x|z)\right)}_{\text{reconstruction term}} - \underbrace{KL\left(q_\phi(z|x)\,\|\,p(z)\right)}_{\text{prior matching term}}\right\} \quad \text{(VAE)}}$$

$q_\phi(z|x)$: Learnable encoder; $v_\theta(x|z)$: Learnable decoder; $p(z)$: prior

Remark Combine (VAE) and (RM) we have

$$\log p_\theta(x) - KL\left(q_\phi(z|x)\,\|\,p_\theta(z|x)\right)$$
$$= \mathbb{E}_{q_\phi(z|x)}\left(\log p_\theta(x|z)\right) - KL\left(q_\phi(z|x)\,\|\,p(z)\right).$$

The LHS is exactly what we want to maximize:

- We want to seek for θ maximizes $p_\theta(x)$
- We want to minimize the encoder q_ϕ and the true posterior p_θ.

With the optimization problem (VAE), what to train in practice?

- Prior matching term: We typically choose a parametric model for $q_\phi(z|x)$ and the prior $p(z)$. A common choice is

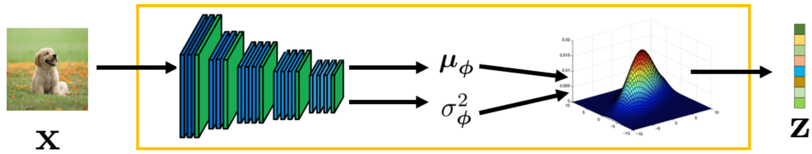
$\boxed{\text{Encoder:}}$ $q_\phi(z|x) = \mathcal{N}(z; \overbrace{\underline{\mu_\phi(x)}, \underline{\sigma_\phi^2(x)}}^{\text{MLP}} I)$

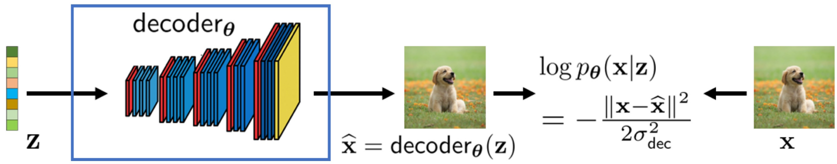
$\boxed{\text{Prior:}}$ $p(z) = \mathcal{N}(z; 0, I)$.

For Gaussian distributions, the explicit form of the KL-divergence is avaliable.

- Reconstruction term: First of all, we will learn a deterministic function through neural network as the decoder function $p_\theta(x|z)$

$\boxed{\text{Decoder:}}$ mean of p_θ $\Leftarrow$ MLP





Then by Monte-Carlo Simulation we can sample $z^{(i)} \overset{i.i.d}{\sim} q_\phi(z|x)$ and estimate the reconstruction term by

$$\frac{1}{n}\sum_{i=1}^{n} \log p_\theta(x|z^{(i)}).$$

However, $z^{(i)}$'s are intrackable with respect to ϕ because they are random samples. Therefore, we typically use the reparametrization trick:

$$z^{(i)} = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon^{(i)},$$

where $\varepsilon^{(i)} \overset{i.i.d.}{\sim} N(0, I)$. Now $z^{(i)}$'s are represented as function of ϕ.

Given training set $\{x^{(\ell)}\}_{\ell=1}^{L}$, initialise ϕ_0, θ_0

for iteration $t \in [0, 1, \cdots, T]$ **do**

Sample mini-batch $\mathcal{D} = \{x^{(\ell_1)}, \cdots, x^{(\ell_K)}\} \subset \{x^{(\ell)}\}_{\ell=1}^{L}$

Sample $z^{(\ell_i)} = \mu_{\phi_t}(x^{(\ell_i)}) + \sigma_{\phi_t}(x^{(\ell_i)}) \odot \varepsilon^{(i)}$, $i = 1, 2, \cdots, K$

Compute: $\frac{1}{K} \sum_{i=1}^{K} \log p_{\theta_t}(x^{(\ell_i)} / z^{(\ell_i)})$

Update θ_{t+1} and ϕ_{t+1} by backpropagating the gradients of (VAE).

end for

1.3 Hierarchical VAE

A hierarchical VAE is a generalization of a VAE that extends to multiple layers of latent variables, i.e. higher level latent variables are permitted.

Let the joint distribution of $(x, Z_{1:T})$ and the posterior distribution (encoder) be

$$p_\theta(x, Z_{1:T}) = p(Z_T)\, p_\theta(x / Z_1) \prod_{t=2}^{T} p_\theta(Z_{t-1} / Z_t)$$

$$q_\phi(Z_{1:T} / x) = q_\phi(Z_1 / x) \prod_{t=2}^{T} q_\phi(Z_t / Z_{t-1}).$$

The the ELBO objective can be derived as

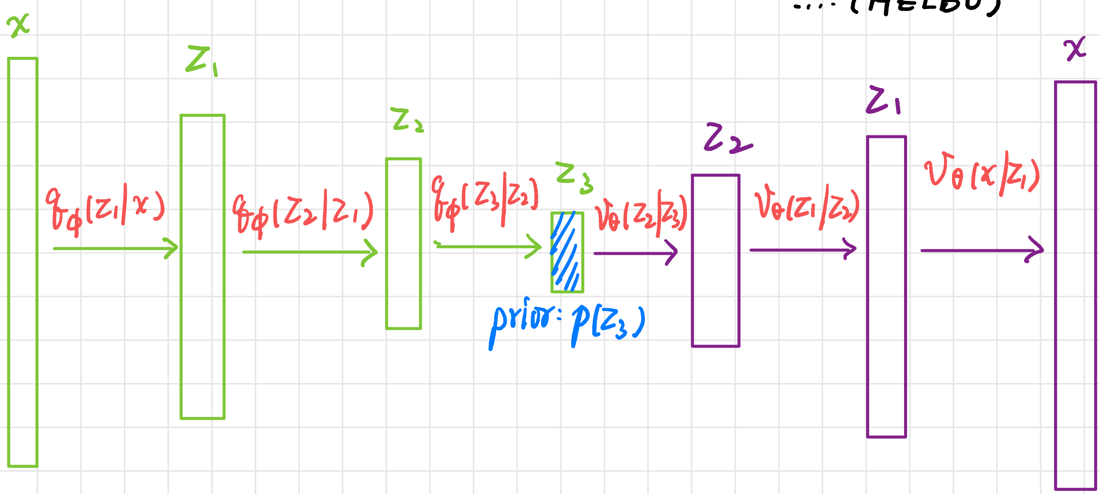
$$\log p_\theta(x) = \log \int p_\theta(x, z_{1:T})\, dz_{1:T}$$

$$= \log \int \frac{p_\theta(x, z_{1:T})\, q_\phi(z_{1:T}|x)}{q_\phi(z_{1:T}|x)}\, dz_{1:T}$$

$$\overset{\text{(Jensen)}}{\geq} \underbrace{\boxed{\mathbb{E}_{q_\phi(z_{1:T}|x)} \left(\log \frac{p_\theta(x, z_{1:T})}{q_\phi(z_{1:T}|x)} \right)}}_{\text{ELBO}}$$

$$= \mathbb{E}_{q_\phi(z_{1:T}|x)} \left(\log \frac{\overbrace{p(z_T)}^{\text{prior}} \overbrace{p_\theta(x|z_1) \prod_{t=2}^{T} p_\theta(z_{t-1}|z_t)}^{\text{decoder}}}{\underbrace{q_\phi(z_1|x) \prod_{t=2}^{T} q_\phi(z_t|z_{t-1})}_{\text{encoder}}} \right)$$

.... (HELBO)



1.4. DDPM.

The DDPM can be seen as a special case of HVAE with the following restrictions:

① The latent dimension is the same as the data dimension. Thus we will simply use x_0 to denote the original data and x_t, $t \geq 1$ to denote the t-th layer of latent variable.

② The encoder is not learned; it is predefined by a Gaussian transition model, i.e.

Encoder

$$q(x_t \mid x_{t-1}) = N(x_t; \mu_t(x_t), \Sigma_t(x_t)),$$

where $\mu_t(x) := \sqrt{\alpha_t}\, x_{t-1}$, $\Sigma_t(x_t) = (1-\alpha_t) I$

By reparametrization trick, we have the forward step:

Forward step:

$$\boxed{x_t = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon, \quad \varepsilon \sim N(0, I)}$$

③ The distribution of the latent at the final step T, i.e. the prior distribution $p(x_T)$, is $N(x_T; 0, I)$

To learn the backward step $p_\theta(x_{t-1}|x_t)$, we derive the ELBO:

$$\log p_\theta(x) \geq \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\,p_\theta(x_0|x_1)\prod_{t=2}^{T} p_\theta(x_{t-1}|x_t)}{q(x_T|x_{T-1})\prod_{t=1}^{T-1} q(x_t|x_{t-1})}\right)$$

different from (HELBO)
the encoder is not learnable q_ϕ.

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\,p_\theta(x_0|x_1)\prod_{t=1}^{T-1} p_\theta(x_t|x_{t+1})}{q(x_T|x_{T-1})\prod_{t=1}^{T-1} q(x_t|x_{t-1})}\right)$$

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\,p_\theta(x_0|x_1)}{q(x_T|x_{T-1})}\right) + \sum_{t=1}^{T-1} \mathbb{E}_{q(x_{1:T}|x_0)}\left(\frac{p_\theta(x_t|x_{t+1})}{q(x_t|x_{t-1})}\right)$$

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log p_\theta(x_0|x_1)\right) + \mathbb{E}_{q(x_{T-1},x_T|x_0)}\left(\log \frac{p(x_T)}{q(x_T|x_{T-1})}\right) + \sum_{t=1}^{T-1} \mathbb{E}_{q(x_{t-1},x_t,x_{t+1}|x_0)}\left(\frac{p_\theta(x_t|x_{t+1})}{q(x_t|x_{t-1})}\right)$$

$$= \overbrace{\mathbb{E}_{q(x_1|x_0)}\left(\log p_\theta(x_0|x_1)\right)}^{\text{reconstruction term}} - \overbrace{\mathbb{E}_{q(x_{T-1}|x_0)}\left[KL\left(q(x_T|x_{T-1})\,\|\,p(x_T)\right)\right]}^{\text{prior matching term}} - \overbrace{\sum_{t=1}^{T-1}\mathbb{E}_{q(x_{t-1},x_{t+1}|x_0)}\left[KL\left(\underbrace{q(x_t|x_{t-1})}_{\text{forward}}\,\|\,\underbrace{p_\theta(x_t|x_{t+1})}_{\text{backward}}\right)\right]}^{\text{consistency term}}$$

Thus the training of the backword step is performed by maximizing the ELBO:

$$\arg\max_{\theta} \Big\{ \mathbb{E}_{q(x_1|x_0)}\left(\log p_\theta(x_0|x_1)\right)$$
$$- \mathbb{E}_{q(x_{T-1}|x_0)}\left[KL\left(q(x_T|x_{T-1}) \,\|\, p(x_T)\right)\right]$$
$$- \sum_{t=1}^{T-1} \mathbb{E}_{q(x_{t-1},x_{t+1}|x_0)}\left[KL\left(q(x_t|x_{t-1}) \,\|\, p_\theta(x_t|x_{t+1})\right)\right] \Big\}$$

$$\cdots\cdots \quad (\text{DDPM})$$

Remark:

a) The reconstruction term maximizes the log-likelihood of the original data given first layer latent.

b) The prior matching term is minimized when the final latent distribution matches the prior.

c) The consistency term endeavors to make the distribution at x_t consistent, from both forward and backforward process.

However, the empirical estimate of (DDPM) often suffers from high variance due to the consistency term is taking expectation on two random variable. Therefore we try another formulation in practical usage.

- A low-variance reformulation of (DDPM)

$$\log p_\theta(x) \geq \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\, p_\theta(x_0|x_1) \prod_{t=2}^{T} p_\theta(x_{t-1}|x_t)}{q(x_T|x_{T-1}) \prod_{t=1}^{T-1} q(x_t|x_{t-1})}\right)$$

same as before

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\, p_\theta(x_0|x_1) \prod_{t=2}^{T} p_\theta(x_{t-1}|x_t)}{q(x_1|x_0) \prod_{t=2}^{T} q(x_t|x_{t-1})}\right)$$

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\, p_\theta(x_0|x_1)}{q(x_1|x_0)} + \log \prod_{t=2}^{T} \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right)$$

(Bayes rule + Markov property)

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\, p_\theta(x_0|x_1)}{q(x_1|x_0)} + \log \prod_{t=2}^{T} \frac{p_\theta(x_{t-1}|x_t)}{\frac{q(x_{t-1}|x_t, x_0)\, \cancel{q(x_t|x_0)}}{\cancel{q(x_{t-1}|x_0)}}}\right)$$

(telescope sum)

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\, p_\theta(x_0|x_1)}{\cancel{q(x_1|x_0)}} + \log \frac{\cancel{q(x_1|x_0)}}{q(x_T|x_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)}\right)$$

$$= \mathbb{E}_{q(x_{1:T}|x_0)}\left(\log \frac{p(x_T)\, p_\theta(x_0|x_1)}{q(x_T|x_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)}\right)$$

$$= \mathbb{E}_{q(x_1|x_0)}\left(\log p_\theta(x_0|x_1)\right)$$

$$+ \mathbb{E}_{q(x_T|x_0)}\left(\log \frac{p(x_T)}{q(x_T|x_0)}\right)$$

$$+ \sum_{t=2}^{T} \mathbb{E}_{q(x_t, x_{t-1}|x_0)}\left(\log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)}\right)$$

$$= \overbrace{\mathbb{E}_{q(x_1|x_0)}\left(\log p_\theta(x_0|x_1)\right)}^{\text{reconstruction term}}$$

$$- \overbrace{KL\left(q(x_T|x_0) \,\|\, p(x_T)\right)}^{\text{prior matching term}}$$

$$- \overbrace{\sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)}\left[KL\left(\underbrace{q(x_{t-1}|x_t, x_0)}_{\text{underlying true backward}} \,\|\, \underbrace{p_\theta(x_{t-1}|x_t)}_{\text{backward decoder}}\right)\right]}^{\text{denoising matching term.}}$$

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

Thus by considering the ELBO we have

$$q(x_t) = \int q(x_t|x_0)\,q(x_0)\,dx_0$$

the equivalent form of (DDPM):

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

$$
\arg\max_{\theta} \Bigg\{ \overbrace{\mathbb{E}_{q(x_1|x_0)}\big(\log p_\theta(x_0|x_1)\big)}^{L_0}
- \overbrace{KL\big(q(x_T|x_0)\,\|\,p(x_T)\big)}^{L_T}
- \overbrace{\sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)}\Big(\underbrace{KL\big(q(x_{t-1}|x_t, x_0)\,\|\,p_\theta(x_{t-1}|x_t)\big)}_{L_{t-1}}\Big)}^{L_{VLB}\text{ (dominate term)}} \Bigg\}
$$

$$
:= \arg\max_{\theta} \sum_{t=1}^{T} L_t . \qquad \cdots\text{(DDPM-LV)}
$$

Now, we are ready to discuss training based on the above.

The following component is needed:

① Forward step: $q(x_t|x_{t-1})$, $t = 2, \cdots, T$

② For any $t \in [T]$: $q(x_t|x_0)$

③ Backward step: $q(x_{t-1}|x_t, x_0)$, $t = 2, \cdots, T$.

④ Prior distribution: $p(x_T)$

1.5. Training: Leverage Gaussian kernel

For arbitrary posteriors in (DDPM-LV), the KL-diverge can be difficult to minimize.

Fortunately, we can leverage the Gaussian transition assumption to make the KL-divergence tractable.

Recap that

① Forward step: $q(x_t | x_{t-1}) = \mathcal{N}(x_t ; \sqrt{\alpha_t}\, x_{t-1}, (1-\alpha_t) I)$,

and by the Bayes rule we have the true transition: ②

Backward step: $q(x_{t-1} | x_t, x_0) = q(x_t | x_{t-1}, x_0) \cdot \dfrac{q(x_{t-1} | x_0)}{q(x_t | x_0)}$, ...(BS)

important to depend on x_0, $q(x_{t-1}|x_t)$ is intractable

$= q(x_t | x_{t-1})$ [Markov]

We only need to find $q(x_t | x_0)$. In fact, we only need note

$$
\begin{aligned}
x_t &= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\ \varepsilon_{t-1} \\
&= \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_{t-1}}\ \varepsilon_{t-2}\right) + \sqrt{1-\alpha_t}\ \varepsilon_{t-1} \\
&= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{\alpha_t - \alpha_{t-1}\alpha_t}\ \varepsilon_{t-2} + \sqrt{1-\alpha_t}\ \varepsilon_{t-1} \\
\left(\text{By } \varepsilon_t \overset{i.i.d}{\sim} N(0,I)\right) &= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{\sqrt{\alpha_t - \alpha_{t-1}\alpha_t}^{\,2} + \sqrt{1-\alpha_t}^{\,2}}\ \varepsilon_{t-2} \\
&= \sqrt{\alpha_t \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_{t-1}\alpha_t}\ \varepsilon_{t-2}
\end{aligned}
$$

$$= \cdots\cdots$$

$$= \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon_0 \quad \cdots\cdots (NP)$$

where $\bar{\alpha}_t = \alpha_t \alpha_{t-1} \cdots \alpha_1$. thus

2

$$\boxed{\text{For any } t\in[T]:\ q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t) I).}$$

By taking this into (BS), we have the true backward transition is (derivation see Appendix)

4

$$\boxed{\text{Backward step: } q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \mu_q(x_t, x_0), \Sigma_q(t)),}$$

$$\cdots (TBW)$$

where

$$\mu_q(x_t, x_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)x_0}{1-\bar{\alpha}_t} \quad \cdots\cdots (TBW\text{-}M)$$

$$\Sigma_q(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \cdot I$$

Now we are ready to discuss training.

Training of $p_\theta(x_{t-1} \mid x_t)$:

Since the underlying true backward transition is Gaussian, it is reasonable to assume $p_\theta(x_{t-1} \mid x_t)$ as a Gaussian distribution as well.

Suppose

$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \overset{\text{MLP}}{\mu_\theta(x_t,t)}, \Sigma_\theta(x_t,t)). \quad \cdots\cdots \text{(LBW)}$$

We first fix

$$\Sigma_\theta(x_t,t) = \Sigma_q(t) \rightarrow \text{known.}$$

and have the network learns only the mean.

Comparison:

True: $q(x_{t-1}|x_t, x_0) = N(x_{t-1}; \underbrace{\mu_q(x_t,x_0)}^{\text{known}}, \underbrace{\Sigma_q(t)}^{\text{known}})$

$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad //$

Train: $p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \underbrace{\mu_\theta(x_t,t)}^{\text{trainable MLP}}, \underbrace{\Sigma_\theta(x_t,t)}^{\text{known}}).$

By (TBW-M), we can set $\mu_\theta(x_t,t)$ to be

$$\mu_\theta(x_t,t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\overset{\text{MLP predict } x_0}{\hat{x}_\theta(x_t,t)}}{1-\bar{\alpha}_t},$$

where $\hat{x}_\theta(x_t,t)$ is a neural network to predict x_0 from the noisy image x_t [Denoising!].

By leveraging the explicit form of KL-divergence between two Gaussian distributions [see, equation (86) in Luo (2021)], and the exact distributions of $\overset{(\text{TBW})}{q(x_{t-1} \mid x_t, x_0)}$ and $\overset{(\text{LBW})}{p_\theta(x_{t-1} \mid x_t)}$, we have

for fixed x_t,

$$\arg\min_\theta KL\left(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t)\right)$$

$$= \arg\min_\theta KL\left(N(x_{t-1}; \mu_q, \Sigma_q(t)) \,\|\, N(x_{t-1}; \underbrace{\mu_\theta}_{\text{MLP}}, \Sigma_q(t))\right)$$

$$= \boxed{\arg\min_\theta \frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2} \| \hat{x}_\theta(x_t, t) - x_0 \|_2^2}$$

where $\sigma_q^2(t) := \dfrac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$. Please find the detailed derivation in Luo (2021) pp.13 and Chan (2024) pp. 23.

Therefore maximizing (DDPM-LV) can be approximated by minimizing the follow:

$$\boxed{\arg\min_\theta \frac{1}{T} \sum_{t=1}^{T} \left[\mathbb{E}_{q(x_t \mid x_0)} \left[\lambda(t) \| \hat{x}_\theta(x_t, t) - x_0 \|_2^2 \right] \right]}, \quad \cdots (\text{P1})$$

where $\lambda(t) = \dfrac{1}{2\sigma_q^2(t)} \dfrac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2}$.

DDPM: Training:

Given training set $\mathcal{D}$, number of iterations H.

for $k \in [0, 1, \cdots, H]$ **do**

Draw x_0 from $\mathcal{D}$

for $i \in [0, 1, \cdots, N]$ **do**

Sample $t \sim \text{Unif}[1, T]$

Sample $x_t \sim q(x_t | x_0) = \mathcal{N}(x_t ; \sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t) I)$

$$x_t = \bar{\alpha}_t x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, I)$$

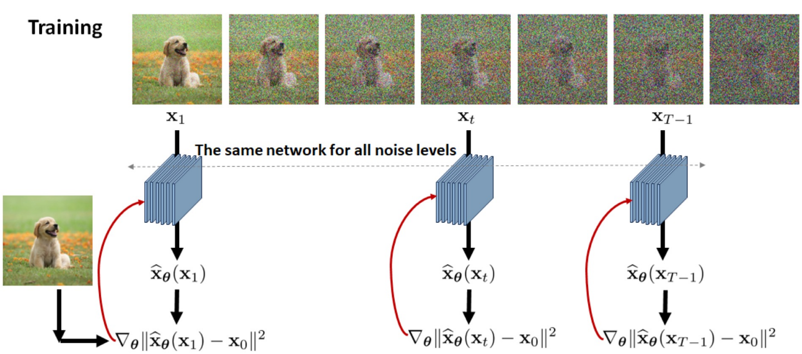
Take gradient descent step on

$$\nabla_\theta \| \hat{x}_\theta(x_t, t) - x_0 \|^2$$

end for

Update θ

end for



With trained $\hat{x}_\theta$, we have

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)).$$

$$\Rightarrow \quad x_{t-1} = \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\bar{\alpha}_t} x_t + \frac{(1-\alpha_t)\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t} \hat{x}_\theta(x_t, t) + \sigma_q(t)\, \varepsilon_t$$

$$\cdots\cdots \text{(Inf)}$$

DDPM: Inference

Given trained $\hat{x}_\theta$ and a white noise $x_T \sim \mathcal{N}(0, I)$

for $t \in [T, T-1, \cdots, 1]$ **do**

Calculate $\hat{x}_\theta(x_t, t)$

Update x_t according to (Inf)

end for

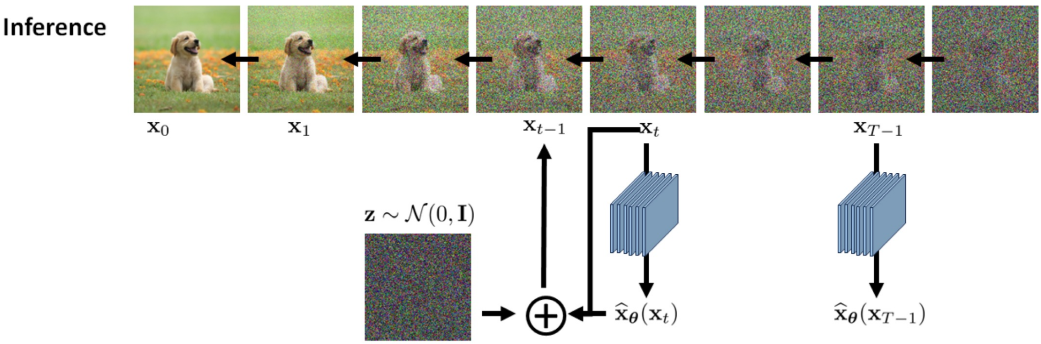


Figure 16: Inference of a denoising diffusion probabilistic model.

- Correctness of DDPM. [Nakkiran et al. (2024) pp. 9.]

In the previous discussion, we use Gaussian model to estimate $p_\theta(x_{t-1} | x_t)$ because $q(x_{t-1} | x_t, x_0)$ is a Gaussian distribution by derivation. However, we are actually interested to model $q(x_{t-1} | x_t)$, i.e. the underlying true backward distribution, then does $q(x_{t-1} | x_t)$ close to Gaussian?

What we estimate :	What we are interested in:
$q(x_{t-1} \| x_t, x_0)$ [Gaussian]	$q(x_{t-1} \| x_t)$
$\mathbb{E}(x_{t-1} \| x_t, x_0)$	$\mathbb{E}(x_{t-1} \| x_t)$

1.6 Equivalent perspectives.

From the derivation of (P1) we see that the DDPM can be interpreted as learning a neural network to predict the original image x_0 given a noisy image x_t. In this Section, we consider two other interpretations.

1.6.1. Random error estimation

Recall the underlying true mean value of the backward transition is given in (TBW-M) as:

$$\mu_q(x_t, x_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)x_0}{1-\bar{\alpha}_t}.$$

$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon_t$

Leveraging the nice property (NP), we have

$$x_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_t}{\sqrt{\bar{\alpha}_t}}$$

Taking this into the form of $\mu_q(x_t, x_0)$

we have

$$\mu_q(x_t, x_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t) \cdot \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_t}{\sqrt{\bar{\alpha}_t}}}{1-\bar{\alpha}_t}$$

$$= \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} + \frac{(1-\alpha_t)}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} \right) x_t - \frac{(1-\alpha_t)\sqrt{1-\bar{\alpha}_t}\,\varepsilon_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}$$

$$= \frac{\alpha_t(1-\bar{\alpha}_{t-1}) + (1-\alpha_t)}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)} x_t - \frac{(1-\alpha_t)\sqrt{1-\bar{\alpha}_t}\,\varepsilon_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}$$

$$= \frac{x_t}{\sqrt{\alpha_t}} - \frac{(1-\alpha_t)\varepsilon_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}.$$

There we can set $\mu_\theta(x_t, t)$ to be

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} x_t - \frac{(1-\alpha_t)}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}} \hat{\varepsilon}_\theta(x_t, t),$$

where $\hat{\varepsilon}_\theta(x_t, t)$ is a neural network to estimate ε_t given the noisy image x_t.

Taking the new form of $\mu_\theta(x_t, t)$ into the explicit form of KL-divergence we have

$$\boxed{\begin{aligned}&\arg\min_\theta KL\left(q(x_{t-1}|x_t, x_0) \| p_\theta(x_{t-1}|x_t)\right)\\ &= \arg\min_\theta \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t} \left[\| \epsilon_t - \hat{\epsilon}_\theta(x_t, t) \|_2^2\right].\end{aligned}}$$

random error estimation

Thus maximizing (DDPM-LV) is almost equivalent to the following optimization:

$$\boxed{\arg\min_\theta \frac{1}{T} \sum_{t=2}^{T} \lambda(t) \mathbb{E}_{q(x_0, x_t)} \| \epsilon_t - \hat{\epsilon}_\theta(x_t, t) \|_2^2}$$

Ho et al. (2015) use this "noise prediction" formula and empirically outperforms the previous "signal prediction" formula.

1.6.2. Score function estimation

In this section, we will utilize Tweedie's formula [See Efron (2011)] throughout the analysis.

Mathematically, for a Gaussian variable $Z \sim N(z; \mu, \Sigma)$, Tweedie's formula states that

$$\mathbb{E}[\mu \mid Z] = \overset{p\times 1}{Z} + \overset{p\times p}{\Sigma} \cdot \overset{p\times 1}{\nabla \log \underbrace{p(Z)}_{\text{marginal of } Z}} \quad \cdots\cdots \text{(TW)}.$$

Recalling that

$$q(x_t \mid \underbrace{x_0}_{\substack{\text{equivalent}\\ \text{to given } \mu}}) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t) I),$$

thus (TW) implies that

$$\mathbb{E}(\overset{\sqrt{\bar{\alpha}_t}\, x_0}{\overset{\shortparallel}{\mu_{x_t}}} \mid x_t) = x_t + (1-\bar{\alpha}_t) \cdot \nabla \log p(x_t),$$

thus the RHS of the above can be seen as an estimator of $\mu_{x_t} = \sqrt{\bar{\alpha}_t}\, x_0$. Therefore

$$\sqrt{\bar{\alpha}_t}\, x_0 \approx x_t + (1-\bar{\alpha}_t)\cdot \nabla \log p(x_t)$$

$$\Downarrow$$

$$x_0 \approx \frac{x_t + (1-\bar{\alpha}_t)\cdot \nabla \log p(x_t)}{\sqrt{\bar{\alpha}_t}}$$

Taking this into $\mu_q(x_t, t)$, we have

$\mathbb{E}(x_{t-1} \mid x_t, x_0)$

$$\mu_q(x_t, t) = \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}\,(1-\alpha_t)x_0}{1-\bar{\alpha}_t}$$

$$\approx \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}\,(1-\alpha_t)\cdot \frac{x_t + (1-\bar{\alpha}_t)\cdot \nabla \log p(x_t)}{\sqrt{\bar{\alpha}_t}}}{1-\bar{\alpha}_t}$$

$$= \frac{1}{\sqrt{\alpha_t}} x_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}} \nabla \log p(x_t)$$

Therefore we can set $\mu_\theta(x_t, t)$ to be

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} x_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}} S_\theta(x_t, t),$$

where $S_\theta(x_t, t)$ is a neural network to estimate the score function $\nabla \log p(x_t)$. Then the

Corresponding optimization problem becomes

$$\boxed{\begin{aligned}&\arg\min_{\theta} D_{KL}\left(q(x_{t-1}|x_t, x_0) \| p_\theta(x_{t-1}|x_t)\right)\\ &= \arg\min_{\theta} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{\alpha_t} \left[\| s_\theta(x_t, t) - \nabla \log p(x_t) \|_2^2\right].\end{aligned}}$$

estimate the score of a noisy image.

Thus the final optimization problem becomes.

$$\boxed{\arg\min_{\theta} \frac{1}{T}\sum_{t=2}^{T} \lambda(t)\, \mathbb{E}_{q(x_t|x_0)} \left[\| s_\theta(x_t, t) - \nabla \log p(x_t) \|_2^2\right]}$$

$$\cdots\cdots (\text{DDPM-SM}).$$

where $\lambda(t) = \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2}{\alpha_t}$.

Remark: $\nabla \log p(x_t)$ is intractable, because the underlying true marginal distribution of x_t is unknown.

Here we introduce score maching trick to solve this problem.

Score matching trick

Note we have the following identity

$$\nabla \log p(x_t) = \frac{\nabla_{x_t} p(x_t)}{p(x_t)}$$

$$= \frac{\nabla_{x_t} \int p(x_t | x_0) p(x_0) dx_0}{p(x_t)}$$

$$= \frac{\int \nabla_{x_t} p(x_t | x_0) p(x_0) dx_0}{p(x_t)}$$

$$= \frac{\int \frac{\nabla_{x_t} p(x_t | x_0)}{\frac{p(x_0, x_t)}{p(x_0)}} p(x_0, x_t) \, dx_0}{p(x_t)}$$

$$= \frac{\int \frac{\nabla_{x_t} p(x_t | x_0)}{p(x_t | x_0)} p(x_0, x_t) \, dx_0 .}{p(x_t)}$$

$$= \frac{\int \nabla_{x_t} (\log p(x_t | x_0)) \, p(x_0, x_t) \, dx_0}{p(x_t)} .$$

$$= \int \nabla_{x_t} (\log p(x_t | x_0)) \, p(x_0 | x_t) \, dx_0$$

$$= \mathbb{E}_{p(x_0 | x_t)} \left[\nabla_{x_t} \log p(x_t | X_0) \right]$$

$$= \mathbb{E}\left[\nabla_{x_t} \log p(x_t \mid x_0) \;\middle|\; x_t \right].$$

We use the following property of conditional expectation.

$$\mathbb{E}[Y \mid U] = \underset{f \in L^2(U)}{\arg\min} \left\{ \mathbb{E} \| Y - f(U) \|_2^2 \right\}.$$

Then we have

$$\nabla \log p(x_t) = \underset{f \in L^2(x_t)}{\arg\min} \left\{ \mathbb{E} \| \nabla_{x_t} \log p(x_t \mid x_0) - f(x_t) \|_2^2 \right\}.$$

<u>Remark</u> $\nabla \log p(x_t \mid x_0)$ is tractable by forward transition.

Thus in order to train a neural network that approximate $\nabla \log p(x_t)$, we consider the following optimization:

$(x_t - 0)$

$$\boxed{\arg\min_{\theta} \frac{1}{T} \sum_{t=2}^{T} \lambda(t)\, \mathbb{E}_{q(x_t \mid x_0)} \left[\| S_\theta(x_t, t) - \nabla_{x_t} \log p(x_t \mid x_0) \|_2^2 \right]}$$

$\cdots\cdots$ (DDPM-C)

2. Score-based generative model

[Mainly based on

Yang Song. Generative Modelling by Estimating Gradients of the Data Distribution. (Blog)

Song and Ermon (2019).]

2.1. Score function

Suppose $x_1, \cdots, x_n \overset{i.i.d.}{\sim} p_\theta(x) = \frac{1}{Z_\theta} e^{-f_\theta(x)}$, the main difficulty of applying MLE based method is that the normalizing constant Z_θ might be intractable. By modelling the score function instead of the density function can sidestep the issue.

Consider the score function

$$\nabla_x \log p_\theta(x).$$

Since $\nabla_x (\log p_\theta(x)) = \nabla_x (-\log Z_\theta - f_\theta(x)) = -\nabla f_\theta(x)$, we don't need to worry about the intractable normalizing constant anymore.

Therefore, we can train score-based models by minimizing the Fisher divergence between the model and the data distributions by

$$\boxed{\min_{\theta} \mathbb{E}_{p(x)}\left[\| \underbrace{s_\theta(x)}_{\text{score network}} - \nabla_x \log p(x) \|_2^2\right]}$$

Although the Fisher divergence is infeasible to compute directly due to the unknown formulation of data score $\nabla_x \log p(x)$, there exists a family of method called score matching that minimizing the Fisher divergence without knowledge of the ground-truth data score.

[See Song and Ermon (2019) for detail].

2.2. Langevin dynamics

Once we obtain a trained score-based model

$$S_\theta(x) \approx \nabla \log p(x),$$

we can use Langevin dynamics to draw sample from $p(x)$. Specifically, it initializes any $x_0 \sim \pi(x)$ some prior distribution, and iterates

$$\boxed{x_{i+1} \leftarrow x_i + \varepsilon \nabla \log p(x_i) + \sqrt{2\varepsilon}\, z_i, \quad i = 1, 2, \cdots, K,} \quad \cdots\cdots (L)$$

where $z_i \sim N(0, I)$. When $\varepsilon \to 0$ and $k \to +\infty$,

$$p_{x_K}(x) \to p(x).$$

Sine $S_\theta(x) \approx \nabla \log p(x)$, we can produce samples by plugging it into (L).

Remarks: For every x, taking gradient of its log-likelihood with respect to x essentially describes what direction in data space to move in order to furthur increase its likelihood. Intuitively, the score function defines a vector field over the support of $p(x)$ pointing to the modes.

Now we can summarize the key idea of the framework of score-based generative modelling:

① Train a score network s.t.

$$S_\theta(x) \approx \nabla \log p(x)$$

② Approximately obtain samples with Langevin dynamics using $S_\theta(x)$.

Three Challenges: [See Song and Ermon (2021)]

a) For low-dimensional data lies in a high-dimensional space, $\nabla \log p(x)$ is ill-defined.

b) The estimation on low-density area is not reliable.

c) For mixture distribution, Langevin dynamics can not correctly recover the weights.

2.3. Score-based generative modeling with multiple noise perturbations

[Song and Ermon (2021)]

Song and Ermon (2021) observes that perturbing data with random Gaussian noise solves all of three challenges:

a) Since the Gaussian noise supports on the whole space, perturbing the original data with a small Gaussian noise will support on the whole space.

b) perturbing with a Gaussian noise with a large variance will raise the probability of the low-density area.

c) perturbing with multiple decreasing level of noise can produce correct sample in relatively small number of steps.

2.3.1. Noise conditional score network (NCSN)

Step 1. Score matching on multiple noise levels.

Take isotropic Gaussian noise $z_i \sim N(0, \sigma_i)$ such that

$$\sigma_1 < \sigma_2 < \cdots < \sigma_L.$$

Then we perturb the original data distribution to obtain the noise-perturbed distribution:

$$p_{\sigma_i}(\tilde{x}) = \int p(x) N(\tilde{x}; x, \sigma_i I) dx$$

equivalently speaking, we have

$$\boxed{\tilde{x}_i = x_0 + \sigma_i z , \quad i = 1, 2, \cdots, L,}$$

where $z \sim N(0, I)$.

The objective is to seek for a Noise conditional score-based Network (NCSN) by minimizing

$$\boxed{\arg\min_{\theta} \sum_{i=1}^{L} \lambda(i) \mathbb{E}_{p_{\sigma_i}(\tilde{x})} \left[\| s_\theta(\tilde{x}, i) - \nabla \log p_{\sigma_i}(\tilde{x}) \|_2^2 \right]}$$

However, the above optimization problem is actually intractable because the underlying true data distribution $p(x)$ is unknown. Fortunately, alternative techniques known as score matching have been proposed to approximate the solution.

Denoising score matching [see Song and Ermon (2019)]

Instead of approximate the score function of the ground truth data distribution $p(x)$, we consider the noise distribution

$$p_{\sigma_i}(\tilde{x}|x) = N(\tilde{x}; x, \sigma_i I).$$

Fit the noise distribution rather than pertubed distribution.

We consider the new objective

$$\boxed{\begin{aligned} &\arg\min_\theta \sum_{i=1}^{L} \lambda(i)\, \mathbb{E}_{p(x)}\left[\mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)} \| S_\theta(\tilde{x}, i) - \nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x}|x)\|_2^2\right]. \\ &= \arg\min_\theta \frac{1}{L} \sum_{i=1}^{L} \lambda(i)\, \mathbb{E}_{p(x)}\left[\mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)} \left\| S_\theta(\tilde{x}, i) + \frac{\tilde{x} - x}{\sigma_i} \right\|_2^2\right]. \end{aligned}}$$

...... (NCSN)

As shown in [Vincent, A connection between score matching and denoising autoencoders, 2011], the solution of the above $s_\theta(\tilde{x}, i) = \nabla \log p_{\sigma_i}(\tilde{x})$ almost surely.

Step 2. Annealed Langevin dynamics.

Algorithm 1 Annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T$.

```
Initialize x̃_0
for i ← 1 to L do
    α_i ← ε · σ_i^2 / σ_L^2                ▷ α_i is the step size.
    for t ← 1 to T do
        Draw z_t ~ N(0, I)
        x̃_t ← x̃_{t-1} + (α_i / 2) s_θ(x̃_{t-1}, σ_i) + √α_i z_t
    end for
    x̃_0 ← x̃_T
end for
return x̃_T
```

$$\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$$

$$\mathbf{z}_t \sim \mathcal{N}(0, I)$$

$$\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$$

$$\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$$

2.4. Compare with DDPM.

– Training

We recap the score matching interpretation of DDPM:

(DDPM-C):

$$\boxed{\arg\min_{\theta} \frac{1}{T}\sum_{t=2}^{T} \lambda(t)\, \mathbb{E}_{q(x_t|x_0)} \left[\|s_\theta(x_t,t) - \nabla_{x_t} \log p(x_t|x_0)\|_2^2\right]},$$

and the score-based method: $\Updownarrow$ Equivalent!

(NCSN):

$$\boxed{\arg\min_{\theta} \frac{1}{L}\sum_{i=1}^{L} \lambda(i)\, \mathbb{E}_{p(x)}\left[\mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)} \|s_\theta(\tilde{x},i) - \nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x}|x)\|_2^2\right]}$$

They are exactly the same!!

– Sampling

As for the sampling part, both DDPM and score-based method gradually decrese the level of noise, although different techniques are used.

In next section we will make the relationship between these two more explicitly.

3. SDE Method

Recap of previous section:

Recall the basics of the DDPM:

- Forward step:

$$x_t = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon_t$$

- Backward step:

$$x_t = \frac{1}{\sqrt{\alpha_{t+1}}} x_{t+1} + \frac{1-\alpha_{t+1}}{\sqrt{\alpha_{t+1}}} S_\theta(x_{t+1}, t+1) + \sqrt{\frac{(1-\alpha_{t+1})(1-\bar{\alpha}_t)}{1-\bar{\alpha}_{t+1}}}\, \varepsilon_{t+1}$$

where the backward sampling relies on the score

$$\arg\min_\theta \frac{1}{T}\sum_{t=2}^{T} \lambda(t)\, \mathbb{E}_{q(x_t, x_0)}\left[\| S_\theta(x_t, t) - \nabla_{x_t} \log p(x_t|x_0)\|_2^2\right]$$

3.1. DDPM and SMLP in Continuous Case

Question: What if the number of perturbation steps approaches infinity in DDPM forward step or the score-matching step of the score-based method?

- DDPM

By the Euler-Maruyama discretisation, the following Variance preserve (VP) SDE

Wienier process

$$dX_t = -\frac{1}{2}\alpha(t) X_t \, dt + \sqrt{\alpha(t)} \, dW_t$$

coincides with the forward step of DDPM:

$$X_t = \sqrt{\alpha_t} X_{t-1} + \sqrt{1-\alpha_t} \, \varepsilon_t .$$

- Denoising score matching with Langevin Dynamics (SMLD).

Note in the step 1 of SMLD, we consider to perturb the original data x_0 by an increasing sequence of variance, i.e.

$$\tilde{x}_i = x_0 + \sigma_i z_i, \quad i = 1, 2, \cdots, L,$$

where $\sigma_1 < \sigma_2 < \cdots < \sigma_L$. Therefore, this forward step can also be written as

$$\boxed{\tilde{x}_{i+1} = \tilde{x}_i + \sqrt{\sigma_{i+1}^2 - \sigma_i^2}\, \varepsilon_i, \quad i = 0, 1, \cdots, L-1.}$$

where $\tilde{x}_0 = x_0$, $\sigma_0 = 0$. This is the EM discretisation of the following Variance Exploding SDE:

$$dx_t = \sqrt{\frac{d\sigma^2(t)}{dt}}\, dW_t.$$

We have successfully incorporate the DDPM and SMLD in a framework of SDE!

3.2. SDE Perspective

Now we can consider a general SDE process:

$$dX_t = f(X_t, t)\,dt + g(t)\,dW_t,$$

where $f(\cdot, t): \mathbb{R}^d \to \mathbb{R}^d$ is a vector value function called the drift coefficient, $g(t) \in \mathbb{R}$ is called diffusion coefficient.

Following previous discussions, each form of SDE will define a way to add noise perturbation, thus there are numerous ways to define the forward perturbation step.

Reversing the SDE for sample generation

Any SDE has a corresponding reverse SDE, whose closed form is given by

$$dX_t = [f(X_t, t) - g^2(t)\nabla_x \log p_t(X_t)]\,dt + g(t)\,dW_t,$$

$t = T, T-1, \cdots, 0$, and $p_t(x)$ is the marginal density function of X_t.

Thus solving the reverse SDE requires us to known the terminal distribution $p_T(\cdot)$ and score function $\nabla_x \log p_t(\cdot)$. We train a time-dependent score-based model $S_\theta(x,t)$, s.t.

$$S_\theta(x,t) \approx \nabla_x \log p_t(x).$$

The training objective for $S_\theta(x,t)$ is a weigted combination of Fisher divergence, given by

$$\boxed{\arg\min_\theta \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{p_t(\cdot)}\left[\lambda(t) \| \nabla_x \log p_t(x) - S_\theta(x,t)\|_2^2\right]}.$$

Thus the reverse procedure is

$$\boxed{dX_t = [f(X_t,t) - g^2(t) S_\theta(X_t,t)]dt + g(t) dW_t},$$

then the sampling procedure can be carried out by the Euler - Maruyama discretisation method.

Appendix.

Analytic Solution to OU-Process.

Consider the OU-process:

$$dX_t = \kappa(\theta - X_t)\,dt + \sigma\, dW_t,$$

where W_t is the standard Brownian motion and $\kappa > 0$, θ and $\sigma > 0$ are constants.

Solution:

Let $Y_t = X_t - \theta$, then the original OU-process becomes

$$dY_t = dX_t = -\kappa Y_t\, dt + \sigma\, dW_t.$$

It can be seen from above that Y_t have a drift towards 0 with exponential rate κ.

This motivates the change of variable

$$Y_t = e^{-t\kappa} Z_t \Leftrightarrow Z_t = Y_t e^{t\kappa},$$

which leads to

$$dZ_t = \kappa e^{t\kappa} Y_t\, dt + e^{t\kappa}\, dY_t.$$

$$= \kappa e^{t\kappa} Y_t \, dt + e^{t\kappa} \left(-\kappa Y_t \, dt + \sigma \, dW_t \right)$$

$$= 0 + e^{t\kappa} \sigma \, dW_t .$$

$$= \sigma e^{t\kappa} dW_t .$$

The Solution to Z_t can be obtained immediately by involving the Itô - Integral

$$Z_t = Z_s + \sigma \int_s^t e^{\kappa u} \, dW_u .$$

Reversing the change of variable, we have

$$X_t = Y_t + \theta = e^{-t\kappa} Z_t + \theta$$

$$= \theta + e^{-t\kappa} Z_s + \sigma e^{-t\kappa} \int_s^t e^{\kappa u} \, dW_u$$

$$= \theta + e^{-t\kappa} (X_s - \theta) e^{s\kappa} + \sigma \int_s^t e^{-\kappa(t-u)} \, dW_u$$

$$= \theta + (X_s - \theta) e^{-\kappa(t-s)} + \sigma \int_s^t e^{-\kappa(t-u)} \, dW_u .$$

Thus

$$X_t = \theta + (X_s - \theta) e^{-\kappa(t-s)} + \sigma \int_s^t e^{-\kappa(t-u)} \, dW_u .$$

Note

$$\mathbb{E}(X_t | X_s) = \theta + (X_s - \theta) e^{-\kappa(t-s)}$$

$$\begin{aligned}
&\mathrm{Cov}(X_t | X_s, X_{t'} | X_s) \\
&= \mathbb{E}\left[(X_t - \mathbb{E}X_t)(X_{t'} - \mathbb{E}X_{t'}) \,\middle|\, X_s \right] \\
&= \mathbb{E}\left[\left(\sigma \int_s^t e^{-\kappa(t-u)} dW_u\right)\left(\sigma \int_s^{t'} e^{-\kappa(t'-u)} dW_u\right) \right] \\
&= \sigma^2 e^{-\kappa(t+t')} \mathbb{E}\left[\int_s^t e^{\kappa u} dW_u \cdot \int_s^{t'} e^{\kappa v} dW_v \right] \\
&= \sigma^2 e^{-\kappa(t+t')} \mathbb{E}\left[\int_s^t e^{2\kappa u} du \right] \\
&= \frac{\sigma^2}{2\kappa} e^{-\kappa(t+t')} \left(e^{2\kappa \min(t',t)} - 1 \right) \\
&= \frac{\sigma^2}{2\kappa} \left(e^{-\kappa|t-t'|} - e^{-\kappa(t+t')} \right).
\end{aligned}$$

where the penultimate equality follows by the Itô isometry.

There we have the explicity solution to the OU - process

$$X_t = \theta + (X_s - \theta) e^{-\kappa(t-s)} + \frac{\sigma}{\sqrt{2\kappa}} \sqrt{e^{-\kappa|t-s|} - e^{-\kappa(t+s)}} \cdot W_t$$

Euler - Maruyma discretisation

For SDE

$$dX_t = \mu(X_t, t)\,dt + \sigma(X_t, t)\,dW_t$$

with $X_0 = x_0$ and W_t is the Wienier process. If we would like to solve the SDE on interval $[0, T]$. Then the Euler - Maruyama discretisation provides a numerical approximation to the exact solution as follows :

- Partition the interval $[0, T]$ into N equal subintervals:

  $$0 = \tau_0 < \tau_1 < \cdots < \tau_N = T \quad \text{and} \quad \Delta t = T/N.$$

- Set $Y_0 = x_0$.
- Recursively define Y_n as

  $$Y_{n+1} = Y_n + \mu(Y_n, \tau_n)\Delta t + \sigma(Y_n, \tau_n)\Delta W_n,$$

  where $\Delta W_n = W_{\tau_{n+1}} - W_{\tau_n}$.

→ Doucet et al.

<u>Example</u>: Consider the OU-process

$$dX_t = \kappa(\theta - X_t)\,dt + \sigma\,dW_t.$$

The the EM discretisation can be written as

$$X_{n+1} = \kappa\theta\Delta t + (1 - \kappa\Delta t)X_n + \sigma\sqrt{\Delta t}\,\varepsilon_n,$$

where ε_n is a Standard Gaussian r.v..

Example Consider Variance Preserving (VP) SDE:

$$dX_t = -\frac{1}{2}\beta(t) X_t\, dt + \sqrt{\beta(t)}\, dW_t.$$

Then, the EM discretisation of the above is

$$X_{t+\Delta t} = X_t - \frac{1}{2}\beta(t) X_t \Delta t + \sqrt{\beta(t)}\,(W_{t+\Delta t} - W_t)$$

$$= X_t - \frac{1}{2}\beta(t) X_t \Delta t + \sqrt{\beta(t)\Delta t}\, \varepsilon_t$$

$$= \left(1 - \frac{1}{2}\beta(t)\Delta t\right) X_t + \sqrt{\beta(t)\Delta t}\, \varepsilon_t$$

[Taylor expansion] $$\approx \sqrt{1-\beta(t)\Delta t}\, X_t + \sqrt{\beta(t)\Delta t}\, \varepsilon_t$$

Thus

$$X_{t+1} = \sqrt{1-\beta(t)}\, X_t + \sqrt{\beta(t)}\, \varepsilon_t,$$

which is same as the forward step of DDPM.

Example Consider the Variance Exploding SDE:

$$dX_t = \sqrt{\frac{d\sigma^2(t)}{dt}}\, dW_t.$$

Then the corresponding EM discretisation is

$$X_{t+\Delta t} = X_t + \sqrt{\frac{d\sigma^2(t)}{dt}\cdot \Delta t}\; \varepsilon_t$$

$$\approx X_t + \sqrt{\sigma^2(t+\Delta t) - \sigma^2(t)}\ \varepsilon_t$$

Thus

$$X_{t+1} = X_t + \sqrt{\sigma^2(t+1) - \sigma^2(t)} \cdot \varepsilon_t .$$

Itô's Formula:

Let W_t be a Brownian motion and X_t be an Itô process satisfies:

$$dX_t = \mu(X_t, t)\,dt + \sigma(X_t, t)\,dW_t.$$

If $f(x,t) \in C^2(\mathbb{R}^2, \mathbb{R})$, then $Y_t = f(X_t, t)$ is also an Itô process satisfies:

$$dY_t = \frac{\partial f}{\partial t}(X_t, t)\,dt + \frac{\partial f}{\partial x}(X_t, t)\,dX_t + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}(X_t, t)(dX_t)^2,$$

where $(dX_t)^2$ given by: $dt^2 = 0$, $dt\,dW_t = 0$, $(dW_t)^2 = dt$.

It follows that

$$dY_t = \left(\frac{\partial f}{\partial t}(X_t, t) + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}(X_t, t)\cdot\sigma^2 + \frac{\partial f}{\partial x}(X_t, t)\cdot\mu\right)dt + \frac{\partial f}{\partial x}(X_t, t)\cdot\sigma\,dW_t.$$

Or in integral form:

$$Y_t = f(X_0, 0) + \int_0^t \frac{\partial f}{\partial t}(X_u, u) + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}(X_u, u)\cdot\sigma^2(X_u, u) + \frac{\partial f}{\partial x}(X_u, u)\cdot\mu(X_u, u)\,du + \int_0^t \frac{\partial f}{\partial x}(X_u, u)\cdot\sigma(X_u, u)\,dW_u.$$