# VCLAB Diffusion Study

Sookwan Han (link)
Visual Computing LAB (link)

## 1 Noise-Conditional Score Network (NCSN)

*Reference: Generative Modeling by Estimating Gradients of Data Distribution (Song et al.)* [7]

### 1.1 Preliminatries

#### 1.1.1 Generative Models

Prior works on "Generative models" pose limitations

- Likelihood-based methods (VAE, Normalizing Flow, etc.)
  - Requires to model **normalized probability**: Special architecture required
  - Surrogate losses are optimized (e.g., ELBO in VAE)
- Generative Adversarial Nets
  - **Training is unstable** (due to adversarial training)
  - GAN objective (adversarial objective) is **not directly comparable**, and not suitable for evaluation (note: likelihood-model models probability, which is directly comparable)

#### 1.1.2 What is "score"??

Definition: (Stein) score is gradient of logarithmic probability density, i.e.,

$$s(x) = \nabla_x \log p_{\text{data}}(x) \tag{1}$$

where $x \in \mathbb{R}^d$ is a datapoint.

#### 1.1.3 What is Langevin dynamics?

Originally came out to explain dynamics of molecules, which is highly **stochastic**: meaning, there exists noise perturbations during process.

#### 1.1.4 Score-matching [4]

**Goal**: train a network: $s_\theta : \mathbb{R}^d \to \mathbb{R}^d$ that predicts scores at given datapoint, i.e.,

$$\text{Find } \theta \text{ s.t. minimizes } \frac{1}{2}\|s_\theta(x) - \nabla_x \log p_{\text{data}}(x)\|^2 \text{ at any } x \tag{2}$$

i.e., find $\theta^*$ s.t.

$$\theta^* = \arg\min_\theta \mathbb{E}_{x \sim p_{\text{data}}(x)}[\frac{1}{2}\|s_\theta(x) - \nabla_x \log p_{\text{data}}(x)\|^2] \tag{3}$$

which is equivalent to solving computationally expensive problem:

$$\theta^* = \arg\min_\theta \mathbb{E}_{x \sim p_{\text{data}}(x)}[\frac{1}{2}\|s_\theta(x)\|^2 + \text{tr}(\nabla_x s_\theta(x))] \tag{4}$$

which is again equivalent to solving a slightly-less computationally expensive problem, known as **Slice Score-Matching** [8]:

$$\theta^* = \arg\min_\theta \mathbb{E}_{v\sim\mathcal{N}(0,\mathbf{I})}\mathbb{E}_{x\sim p_{\text{data}}(x)}[v^T \nabla_x s_\theta(x) v + \frac{1}{2}\|s_\theta(x)\|^2] \tag{5}$$

**Problem 1.** Show that Eq. 3 and Eq. 4 are equivalent.

**Problem 2.** Show that Eq. 4 and Eq. 5 are equivalent.

## 1.2 Challenges in Score-Based Generative Modelling

### 1.2.1 Manifold Hypothesis & Inaccurate Score Estimation

The space data can lie in is high-dimensional, i.e., $d$ from $x \in \mathbb{R}^d$ is very big. However, dataset only covers a very small subset of the high-dimensional space: *so we can assume dataset lies on low-dimensional manifold*. This causes the following problems when score-matching:

- Data is not sampled from most of the space except manifold (i.e., *ambient space*); hence, **bad estimates of score** in ambient space
- when sampling, if initial datapoint is given in low-density region, not much information is given $\rightarrow$ **Wrong sampling may occur, or sample may not move to the mode (peak of distribution)**

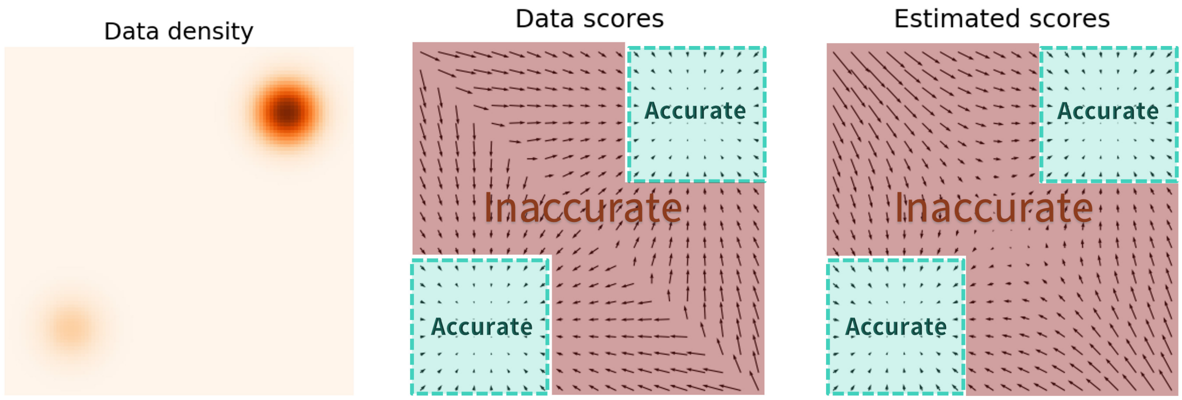


Figure 1: Reference: [6].Inaccurate score estimation in low-density regions

### 1.2.2 Slow mixing of Langevin Dynamics

When sampling with Langevin Dynamics, we require careful steps when moving sample to mode to model correct weights between modes. **I.e., to retrieve correct ratio of samples from different modes (i.e., 1000 samples in region with density $\frac{1}{5}$ and 2000 samples in region with density $\frac{2}{5}$), we need *many "small-step-size" steps* of Langevin steps applied.**
This is because scores tend to get *dominated* with main-factor when near the mode and *ignore* the weights of mode. Look at the simple toy example below:

$$p_{\text{data}}(x) = \pi p_1(x) + (1-\pi)p_2(x) \tag{6}$$

When $x$ is near the mode of $p_1$, the density is dominated by the main-factor $p_1$ as:

$$\text{If } x \text{ near } \arg\max_x p_1(x),\ p_{\text{data}}(x) \simeq \pi p_1(x) \tag{7}$$

Then, the score would be approximately

$$\nabla_x \log p_{\text{data}}(x) \simeq \nabla_x \log p_1(x) + \nabla_x \log \pi = \nabla_x \log p_1(x) \tag{8}$$

which ignores the weighting factor $\pi$. Hence, to take into account the minuscule effect of $\pi$, we require **very small step size** and **many steps** during Langevin sampling.

## 1.3 How does this work mitigate these problems?

### 1.3.1 Perturbing data-distribution with noise

Score-estimation & score-matching was inaccurate in low data-density regions (ambient space) since we can't sample data from these regions to use for training. To mitigate this, we ***intentionally mix noise to the existing data*** to generate samples in the low-density regions. Higher the noise, the more uniform the data will cover the ambient space.
We generate *noisy* samples $\tilde{x}$ from original data samples $x$ using a predefined noise kernel:

$$q_\sigma(\tilde{x}|x) = \mathcal{N}(\tilde{x}|x, \sigma^2\mathbf{I}) \tag{9}$$

Then we can express a marginal distribution for noisy sample $\tilde{x}$:

$$q_\sigma(\tilde{x}) = \int_x q_\sigma(\tilde{x}|x)p_{\text{data}}(x)dx \tag{10}$$

Now we will aim to train a **Noise-Conditioned Score Network** that estimates scores for **noisy distribution** $q_\sigma(\tilde{x})$ for every **noise condition** $\sigma$:

$$\text{Find } \theta \text{ s.t. minimizes } \frac{1}{2}\|s_\theta(\tilde{x};\sigma) - \nabla_x \log p_{\text{data}}(\tilde{x})\|^2 \text{ at any noisy sample } \tilde{x} \text{ obtained via } q_\sigma(\tilde{x}) \tag{11}$$

i.e., find $\theta^*$ s.t.

$$\theta^* = \arg\min_\theta \mathbb{E}_{\tilde{x}\sim q_\sigma(\tilde{x})}[\frac{1}{2}\|s_\theta(\tilde{x};\sigma) - \nabla_{\tilde{x}} \log q_\sigma(\tilde{x})\|^2] \tag{12}$$

This is intractable, as we need to sample from $q_\sigma(\tilde{x})$. However, with a slight mathematical trick, we can make this into an equivalent problem that is tractable:

$$\theta^* = \arg\min_\theta \mathbb{E}_{x\sim p_{\text{data}}(x)}\mathbb{E}_{\tilde{x}\sim q_\sigma(\tilde{x}|x)}[\frac{1}{2}\|s_\theta(\tilde{x};\sigma) - \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x)\|^2] \tag{13}$$

where $q_\sigma(\tilde{x}|x)$ is tractable and $x \sim p_{\text{data}}(x)$ is also tractable as we can replace it with Monte-Carlo sampling.

**Problem 3.** Show that Eq. 12 and Eq. 13 are equivalent.

Using the given form of $q_\sigma(\tilde{x}|x)$ as in Eq. 9 and substituting the components in Eq. 13, we can know derive the objectives:

$$\mathcal{L}(\theta;\sigma) = \frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}(x)}\mathbb{E}_{\tilde{x}\sim\mathcal{N}(x,\sigma^2\mathbf{I})}[\|s_\theta(\tilde{x};\sigma) - \frac{x-\tilde{x}}{\sigma^2}\|^2] \tag{14}$$

for all noise scale $\sigma$:

$$\mathcal{L}(\theta;\{\sigma_i\}_{i=1}^T) = \frac{1}{T}\sum_{i=1}^{T}\lambda(\sigma_i)\mathcal{L}(\theta;\sigma_i) \tag{15}$$

where $\lambda(\sigma_i)$ is a scaling hyperparameter. Since $\frac{x-\tilde{x}}{\sigma} \sim \mathcal{N}(0,\mathbf{I})$, it seems reasonable to set $\lambda(\sigma) = \sigma^2$. This way, the order of magnitude of $\lambda(\sigma)\mathcal{L}(\theta;\sigma)$ does not depend on $\sigma$.

### 1.3.2 Annealed Langevin Dynamics for Inference

When the data-distribution $p_{\text{data}}(x)$ is perturbed by high-noise scale $\sigma$, we can span the whole space and mollify the distribution to have minimal low-density region. This means that the score-estimations will be accurate for any points $x$ when $\sigma$ is high; meaning, for any initialization, the sample can find its way to the mode.
After the NCSN $s_\theta(x;\sigma)$ is trained, we can use it to push samples to the mode of $q_{\sigma_1}(\tilde{x})$ as:

$$\tilde{x}_t \leftarrow \tilde{x}_{t-1} + \frac{\alpha_i}{2}s_\theta(\tilde{x}_{t-1};\sigma_i) + \sqrt{\alpha_i}\mathbf{z_t} \tag{16}$$

where $\mathbf{z}_t$ is Langevin noise, $\alpha_i$ is step-size at noise-scael $\sigma_i$. Then, we can use the final spot at noise $\sigma_1$ as initialization for **reduced noise-scale distribution** $q_{\sigma_2}(\tilde{x})$. Again, we can push the samples and use the final spot as initialization for smaller noise-scale $\sigma_3$.

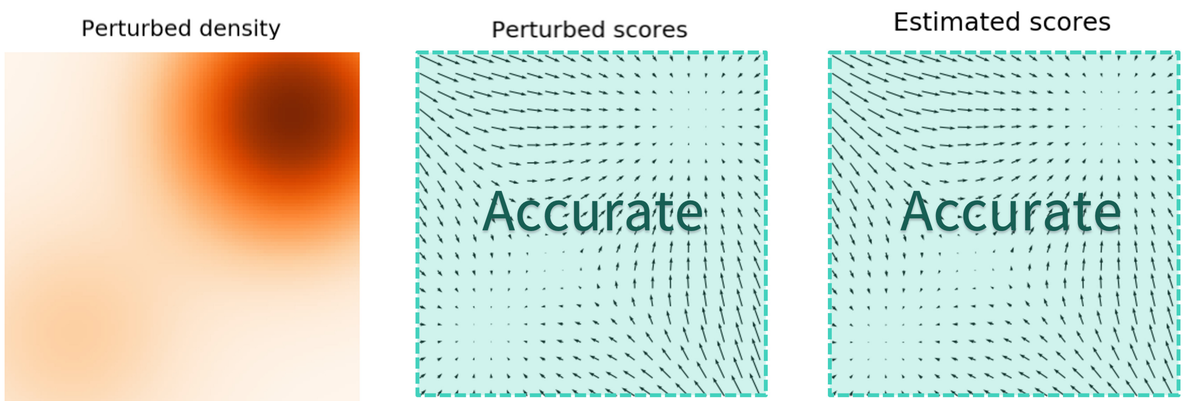


Figure 2: Reference: [6]. Accurate score estimation when noise is perturbed.

As noise-scale becomes smaller ($\frac{\sigma_1}{\sigma_2} = \frac{\sigma_1}{\sigma_2} = ... = \frac{\sigma_{T-1}}{\sigma_T} > 1$)

- score estimations will become more inaccurate in the low-density regions
- slower and more careful Langevin mixing is required

However, as we use the previous sampling results as the initialization (prior) for sampling at current noise-scale,
**1. we do not need to worry about inaccuracy as we will already be in high-density region.**
Also,
**2. By applying progressively smaller step-sizes for sampling as noise-scale reduces, we can mitigate the second problem addressed.**

## 2 Denoising Diffusion Probabilistic Models (DDPM)

*Reference: Denoising Diffusion Probabilistic Models (Ho et al.)* [2]

**Diffusion model takes a probabilistic approach of viewing SDEs. It is later known to be equivalent as SDE.**

### 2.1 Objectives

Diffusion model is a Markov-Chain **latent variable model** that "destroys" training data through addition of Gaussian Noise, and learn how to "denoise" a noisy sample. This way, at inference, we can generate samples from arbitrary noise.
We assume that we are given how the data $x_0 \sim p_{\text{data}}$ is destroyed **(forward kernel)**:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}), \ \ 0 < \beta_t < 1 \tag{17}$$

which can be aggregated to a "shortcut" from $x_0$ to $x_t$:

$$q(x_t|x_0) = \mathcal{N}(x_t|\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}), \ \ \alpha_t = 1-\beta_t, \ \ \bar{\alpha}_t = \alpha_t\alpha_{t-1}...\alpha_1 \tag{18}$$

The aim of the diffusion model is to learn how to "reverse" the noising process (i.e., denoising) starting from the random noise $x_T$ to $x_0$ to sample data $x_0$:

$$\text{Find } \theta \text{ s.t. following processes } p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}|\mu_\theta(x_t;t), \Sigma_\theta(x_t;t)) \text{ gives } x_0 \sim p_{\text{data}}(x_0) \tag{19}$$

i.e., finding $\theta^*$ that maximizes the marginalized probability of $x_0$ for $x_0$ sampled from dataset:

$$\theta^* = \arg\min_\theta \mathbb{E}_{x_0 \sim p_{\text{data}}(x)}[-\log p_\theta(x_0)] = \arg\min_\theta \mathbb{E}_{x_0 \sim p_{\text{data}}(x)}[-\log \int_{x_{1:T}} p_\theta(x_{0:T})dx_{1:T}] \tag{20}$$

**Problem 4.** Derive Eq. 18 from Eq. 17.

**Problem 5.** Assume $\text{Var}(x_0) = 1$. Prove that $\forall t, \text{Var}(x_t) = 1$.

### 2.2 Surrogate Objectives

By Jensen's inequality, we obtain the upper-bound of negative-log-likelihood:

$$\mathbb{E}_{x_0 \sim p_{\text{data}}(x)}[-\log \int_{x_{1:T}} p_\theta(x_{0:T}) dx_{1:T}] \leq \mathbb{E}_{x_0 \sim p_{\text{data}}} \mathbb{E}_{x_{1:T} \sim q(x_{1:T}|x_0)}[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}] \tag{21}$$

Using the Markov Chain property, we can decompose the right-hand side to:

$$\mathcal{L}_{\text{vlb}} := \mathbb{E}_{x_0 \sim p_{\text{data}}} \mathbb{E}_{x_{1:T} \sim q(x_{1:T}|x_0)}[-\log p_\theta(x_T) - \sum_{t=1}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}] \tag{22}$$

The surrogate objective looks good, but holds problem since we need to compare $x_{t-1}$ from $p_\theta(x_{t-1}|x_t)$ and $x_t$ from $q(x_t|x_{t-1})$, which is unsuitable for direct computation (in the form of KL-divergence). Apply Bayes' rule, and we obtain:

$$\mathcal{L}_{\text{vlb}} = \mathbb{E}_{x_0 \sim p_{\text{data}}} \mathbb{E}_{x_{1:T} \sim q(x_{1:T}|x_0)}[-\log \frac{p_\theta(x_T)}{q(x_T|x_0)} - \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} - \log p_\theta(x_0|x_1)] \tag{23}$$

Instead of computing and adding all the terms in Eq. 23 for optimization, we choose to random-sample timestep $t$ and optimize following surrogate objective:

$$\forall t = 2, ..., T: \quad \mathcal{L}_{t-1} = \mathbb{E}_{x_0 \sim p_{\text{data}}} \mathbb{E}_{x_{1:T} \sim q(x_{1:T}|x_0)}[-\log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)}] \tag{24}$$

which is same as KL-divergence of two distributions!

**Problem 6.** Prove Eq. 21

**Problem 7.** Derive Eq. 23 from Eq. 22.

## 2.3 Computing Surrogate Losses

The $x_0 \sim p_{\text{data}}(x)$ term in Eq. 24 is tractable if we replace it with Monte-Carlo sampling; hence, we only need to know how to compute $\mathbb{E}_{x_{1:T} \sim q(x_{1:T}|x_0)}[-\log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)}]$, which is same as

$$D_{\text{KL}}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \tag{25}$$

Note that for two Gaussians, this value is easy to compute:

$$D_{\text{KL}}(p||q) = \frac{1}{2}[\mu_p^T \mu_p + \text{tr}(\Sigma_p) - d - \log|\Sigma_p|] \tag{26}$$

where $d$ is dimension. So basically, **we are trying to fit $p_\theta(x_{t-1}|x_t)$ to $q(x_{t-1}|x_t, x_0)$.** Let's denote $q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0; t), \tilde{\beta}(t)\mathbf{I})$. Then, computing Eq. 26 suffices to computing:

$$\frac{1}{2\sigma_t^2}\|\tilde{\mu}(x_t, x_0; t) - \mu_\theta(x_t; t)\|^2 \tag{27}$$

where $\sigma_t^2 = \tilde{\beta}(t) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$.

**Problem 8.** Prove that $q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0; t), \tilde{\beta}(t)\mathbf{I})$ where:

$$\tilde{\mu}(x_t, x_0; t) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon), \;\; \tilde{\beta}(t) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \tag{28}$$

by applying Bayes' rule to Eq. 17 and Eq. 18. (note: $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$)

We're back again, **let's try to make $\mu_\theta$ as same structure as $\tilde{\mu}$.** Thinking of $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$, since $\tilde{\mu}$ is given up in **Problem 8**, we can re-parametrize ***epsilon term*** to express the mean $\mu_\theta$:

$$\mu_\theta(x_t; t) = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t; t)) \tag{29}$$

Then, we can finally express surrogate objective $\mathcal{L}_{t-1}$ as:

$$\mathcal{L}_{t-1} = \mathbb{E}_{x_0 \sim p_{\text{data}}(x), \epsilon \sim \mathcal{N}(0,\mathbf{I})}[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon; t)\|^2] \tag{30}$$

which is very similar to NCSN!! (except noise-scale $\sigma$ is turned to timescale $t$) The sampling procedure:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t; t)) + \sigma_t\mathbf{z}_t \tag{31}$$

is also very similar to the Langevin dynamics of NCSN ($\epsilon_\theta$ resembles score-function) Note that $x_T \sim \mathcal{N}(0, \mathbf{I})$ is sampled from pure Gaussian noise, assuming the noise-scale $\bar{\alpha}_t$ is almost 1 when $t \to T$.

# 3 Classifier-Free Guidance

*Reference: Classifier-Free Diffusion Guidance (Ho et al.)* [3]

Now we consider the situation where we would like to *condition* the diffusion model with some condition $c$, that is:

$$\epsilon_\theta(x_t;t) \to \epsilon_\theta(x_t,c;t) \tag{32}$$

## 3.1 Classifier Guidance

In *Diffusion beat GANs on Image Synthesis (Dariwhal et al.)* [1], the authors introduce an auxiliary classifier $p_\phi$ to the model as:

$$\tilde{\epsilon}_\theta(x_t,c;t) = \epsilon_\theta(x_t,c;t) - \omega\sigma_t \nabla_{x_t} \log p_\phi(c|x_t) \tag{33}$$

which pushes the sample $x_t$ to the high $p_\phi(c|x_t)$ **(aka, fidelity to given condition)** region when the sample takes a step following Eq. 31, where as $\omega$ increases, fidelity to condition increases but diversity of samples decrease (fidelity-diversity trade-off).

## 3.2 Classifier-Free Guidance

Challenges of **Classifier-Guidance** is:

- Additional classifier-model (additional parameters $\phi$) is required, complicating training pipeline
- Classifier-model needs to be trained on noisy data $x_t$: *we cannot plug-in pretrained model*

Let's try to achieve similar effects without having to define any auxiliary models! $\to$ **Classifier-Free Guidance**

Instead of using classifier model $p_\phi$, let's think of an *implicit classifier* $p(c|x_t)$, which is by Bayes' rule:

$$p(c|x_t) = \frac{p(x_t|c)p(c)}{p(x_t)} \to \log p(c|x_t) = \log p(x_t|c) - \log p(x_t) + C \tag{34}$$

where it is natural to think of design choice of $p(x_t|c)$ and $p(x_t)$ be modelled with $\theta$. Then, to increase the fidelity of samplme $x_t$ w.r.t. condition $c$, we can aim to maximize:

$$p_\theta(x_t)p_\theta(c|x_t)^\omega \tag{35}$$

instead of $p_\theta(x_t)$, where the score-function would be:

$$\nabla_{x_t}[\log p_\theta(x_t)p_\theta(c|x_t)^\omega] = \nabla_{x_t} \log p_\theta(x_t) + \omega(\nabla_{x_t} p_\theta(x_t|c) - \nabla_{x_t} p_\theta(x_t)) \tag{36}$$

This can be re-written in $\epsilon$-notation as in DDPM via simple rescaling:

$$\epsilon_\theta^{\text{new}}(x_t,c;t) = (1+\omega)\epsilon_\theta(x_t,c;t) - \omega\epsilon_\theta(x_t;t) \tag{37}$$

where $\omega$ is known as classifier-free guidance weight. This process resembles of extrapolating the score-vectors *toward the direction of mode-when-conditioned*.

**Problem 9.** What would be the scale-factor between the *real score function* $\nabla_{x_t} \log p_\theta(x_t)$ and *normalized noise perturbation from DDPM* $\epsilon_\theta(x_t)$?

# 4 DDIM

*Reference: Denoising Diffusion Implicit Models (Song et al.)* [5]

## 4.1 Challenges with Diffusion Models

Although diffusion models show good sample quality, the model has some drawbacks:

- **Very slow inference time**. Due to iterative denoising process, the model requires multiple model passes which slows inference process.
- **Uncontrolled stochasticity.** Although the stochastic reverse-process of diffusion model gives us diversity of samples, we cannot control it.

To mitigate these challenges, ***new non-Markovian forward kernel*** that can introduce a new hyperparameter that controls *stochasticity* of generation process.

## 4.2 Non-Markovian Forward Kernel

**RECALL**: In DDPM, the forward kernel was *defined* as:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}) \tag{38}$$

so that we have marginalized kernel:

$$q(x_t|x_0) = \mathcal{N}(x_t|\sqrt{\bar{\alpha}_t}x_{t-1}, (1-\bar{\alpha}_t)\mathbf{I}) \tag{39}$$

which is used for sampling $x_t$ directly from $x_0$ (shortcut!).

We will **re-define** the forward kernel (i.e., DDIM forward kernel) to have same marginalized kernel, but also **stochastically-controllable**, as:

$$q_\sigma(x_T|x_0) = \mathcal{N}(x_T|\sqrt{\bar{\alpha}_T}x_0, (1-\bar{\alpha}_T)\mathbf{I}) \tag{40}$$

$$\text{for } t = 2, ..., T: \quad q_\sigma(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}|\sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2} \cdot \frac{x_t - \sqrt{\bar{\alpha}_t}x_0}{\sqrt{1-\bar{\alpha}_t}},\ \sigma_t^2\mathbf{I}) \tag{41}$$

Note that $\sigma = [\sigma_1, ..., \sigma_T]$ which is an hyperparameter. In practice, the $\sigma$ values are set as $\sigma_t = \eta\sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}}\sqrt{1-\alpha_t}$ where $\eta$ is hyperparameter for controlling the stochasticity of diffusion trajectory. **If $\eta = 0$, we get deterministic diffusion trajectory (i.e., implicit model)**!

**Problem 10.** Prove that marginalizing DDIM forward kernel in Eq. 40 and Eq. 41 gives us the same distribution as in DDPM (hint: use Bayes' rule).

$$q(x_t|x_0) = \mathcal{N}(x_t|\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{42}$$

### 4.3 Variational Inference Objective of DDIM

Similar to DDPM, we can write the variational objective (uppder bound of $\mathbb{E}_{x_0 \sim p_{\text{data}}(x)}[-\log p_\theta(x_0)]$) as:

$$\mathcal{L}_{\text{vlb}}(\theta;\sigma) = \mathbb{E}_{x_0 \sim p_{\text{data}}} \mathbb{E}_{x_{1:T} \sim q_\sigma(x_{1:T}|x_0)}[-\log \frac{p_\theta(x_T)}{q_\sigma(x_T|x_0)} - \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q_\sigma(x_{t-1}|x_t, x_0)} - \log p_\theta(x_0|x_1)] \quad (43)$$

**RECALL**: In DDPM, surrogate objective (of each timestep) was:

$$\mathcal{L}_{t-1} = \mathbb{E}_{x_0 \sim p_{\text{data}}(x), \epsilon_t \sim \mathcal{N}(0,\mathbf{I})}[\|\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t; t)\|^2] \quad (44)$$

The total surrogate objective is defined as:

$$\mathcal{L}_\gamma(\theta) = \sum_{t=1}^{T} \gamma_t \mathcal{L}_{t-1} \quad (45)$$

where $\gamma = [\gamma_1, ..., \gamma_T]$ is a scaling-factor.

Note that this was surrogate objective for DDPM; however, following the **Theorem**, optimizing $\mathcal{L}_\gamma$ can be equivalent to optimizing $\mathcal{L}_{\text{vlb}}(\theta;\sigma)$ for ***ANY*** $\sigma = [\sigma_1, ..., \sigma_T]$ when $\gamma$ satisfies some condition:

**Theorem:** *For all* $\sigma > 0,$ *there exists* $\gamma \in \mathbb{R}^T_{>0}$ *and* $C \in \mathbb{R}$ *such that* $\mathcal{L}_{\text{vlb}}(\theta;\sigma) = \mathcal{L}_\gamma(\theta) + C$ (46)

This means, ***we do not need to train or further finetune any diffusion model!!!***

**Problem 11.** Prove **Theorem**.

### 4.4 Sampling for DDIM

Again, after training $\epsilon_\theta$, we can use it to model the reverse-kernel $p_\theta(x_{t-1}|x_t)$ as below:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t;t)}{\sqrt{\bar{\alpha}_t}}) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\epsilon_\theta(x_t;t) + \sigma_t\mathbf{z}_t \tag{47}$$

Each components have different semantics, and are worth noting the meanings:

- $\sqrt{\bar{\alpha}_{t-1}}(\frac{x_t-\sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t;t)}{\sqrt{\bar{\alpha}_t}})$: Denotes "predicted $x_0$". The reverse process, while it is viewed as predicting $\epsilon_\theta$, can also be viewed as predicting $x_0$ form $x_t$.
- $\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\epsilon_\theta(x_t;t)$: Denotes "direction-of-score at $x_t$". So basically this is the vector-field that pushes the sample towards the mode.
- $\sigma_t\mathbf{z}_t$: Denotes "Langevin noise". This was uncontrollable in DDPM, but by a clever mathematical trick in DDIM (re-defining forward kernel in non-Markovian manner), we can control this stochasticity (e.g., to remove stochasticity, simply set $\sigma = \mathbf{0}$!!)

As mentioned earlier, during practice, the Langevin noise scale $\sigma_t$ is set as $\sigma_t = \eta\sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}}\sqrt{1-\alpha_t}$:

- If $\eta = 0$, the forward-kernel is ***deterministic process***.
- If $\eta = 1$, the forward-kernel is same as in DDPM.

**Problem 12.** Show that when $\eta = 1$, Eq. 47 is equivalent to reverse-process of DDPM.

### 4.5 Accelerated Sampling

If we set $\gamma = \mathbf{1}$, the surrogate objective $\mathcal{L}_\mathbf{1}$ does not depend on the user-defined noise schedule $\sigma$, which means **we can use pretrained diffusion model** for shorter timesteps, as long as:

- Timestep-schedule $[\tau_1, \tau_2, ..., \tau_s]$ is included in the original timestep-schedule $[1, ..., T]$
- $\sigma_{\text{shorter}} = [\sigma_{\tau_1}, \sigma_{\tau_2}, ..., \sigma_{\tau_3}]$

In this case, the marginals become:

$$q(x_{\tau_i}|x_0) = \mathcal{N}(x_{\tau_i}|\sqrt{\bar{\alpha}_{\tau_i}}x_0, (1-\bar{\alpha}_{\tau_i})\mathbf{I}) \tag{48}$$

and sampling equation becomes:

$$x_{\tau_{i-1}} = \sqrt{\frac{\bar{\alpha}_{\tau_i}}{\bar{\alpha}_{\tau_{i-1}}}}(\frac{x_t - \sqrt{1-\bar{\alpha}_{\tau_i}}\epsilon_\theta(x_{\tau_i};\tau_i)}{\sqrt{\bar{\alpha}_{\tau_i}}}) + \sqrt{1-\frac{\bar{\alpha}_{\tau_i}}{\bar{\alpha}_{\tau_{i-1}}}-\sigma_t^2}\cdot\epsilon_\theta(x_t;t) + \sigma_t\mathbf{z}_t \tag{49}$$

which is much faster if $s = \text{len}([\tau_1, ..., \tau_s])$ (i.e., $s = 50$) is way smaller than original DDPM (i.e., 1000 steps)!!

# References


[1] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[3] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[4] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[5] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[6] Yang Song. Generative modeling by estimating gradients of the data distribution. `https://yang-song.net/blog/2021/score/`, 2021.

[7] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[8] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pages 574–584. PMLR, 2020.