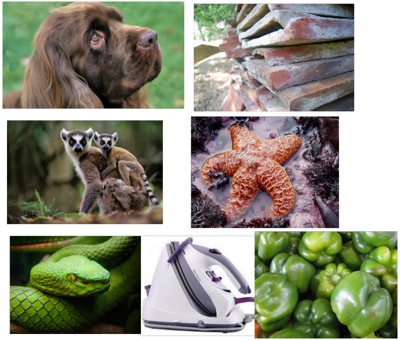
Denoising Diffusion Models

George Deligiannidis

October 23, 2025



(a) Training samples from IMAGENET

(b) Generated samples from a diffusion model.

Figure: Images from Rombach, Robin, et al. "High-resolution image synthesis with latent diffusion models." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022.

AIM: Given access to samples from p_{data}, learn to sample from $p_{\text{model}} \approx p_{\text{data}}$.

Generative Modelling

Numerous applications:

- Image1/video/sound generation (diffusions)
- Protein structure discovery2 (diffusions)
- Time series (diffusions)
- ChatGPT and other AI chatbots also generative models (some diffusion)
- Many successful models: autoregressive models, normalising flows, Generative Adversarial Networks Until recently GANs SOTA

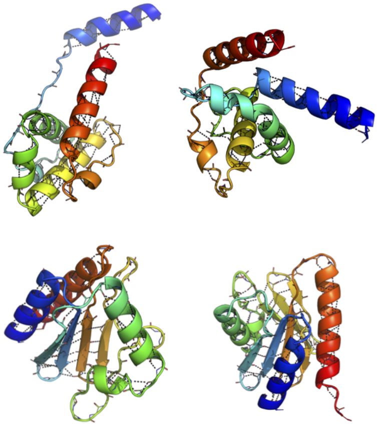


Figure: Training samples from IMAGENET

Introduction to Denoising Diffusion Models

Introduction to Denoising Diffusion Models

A new contender: **Denoising Diffusion Models**

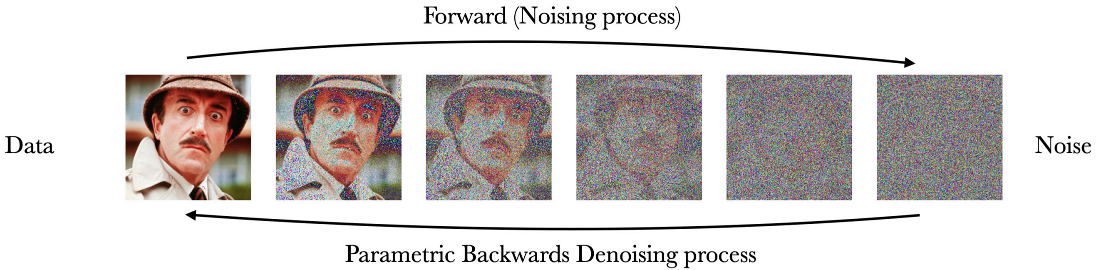
- Advantages:
- State-of-the-art results
- Very flexible
- More amenable to theoretical analysis (than e.g. GANs)

Figure: Generated samples from a diffusion model trained on CelebrA-HQ (faces).[a]

[a] Images from Rombach, Robin, et al. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022.

The main idea

- **Corrupt** data by progressively adding noise, until indistinguishable from noise
- **Learn reverse** denoising process
- **Apply** the reverse denoising process to noise to produce fresh samples



Ingredients 1: score matching

- We are given samples $X_1, \ldots, X_n \sim p_{\mathsf{data}}$; **how to produce more?**
- If we knew the **score function** $\nabla_x \log p_{\mathsf{data}}(x)$: use Langevin MCMC to sample.
- Let's learn the score function from data — **score matching** (SM) (Hyvärinen 2005).

Ingredients 1: score matching

- We are given samples $X_1, \ldots, X_n \sim p_{\text{data}}$; **how to produce more?**
- If we knew the **score function** $\nabla_x \log p_{\text{data}}(x)$: use Langevin MCMC to sample.
- Let's learn the score function from data — **score matching** (SM) (Hyvärinen 2005).

Explicit Score Matching (ESM): $$\operatorname{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \| s_\theta(X_i) - \nabla \log p_{\text{data}}(X_i) \|^2.$$

Ingredients 1: score matching

- We are given samples $X_1, \ldots, X_n \sim p_{\text{data}}$; **how to produce more?**
- If we knew the **score function** $\nabla_x \log p_{\text{data}}(x)$: use Langevin MCMC to sample.
- Let's learn the score function from data — **score matching** (SM) (Hyvärinen 2005).

Explicit Score Matching: $$\operatorname{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \| s_\theta(X_i) - \underbrace{\nabla \log p_{\text{data}}(X_i)}_{\text{intractable}} \|^2.$$

Ingredients 1: score matching

- We are given samples $X_1, \ldots, X_n \sim p_{\text{data}}$; **how to produce more?**
- If we knew the **score function** $\nabla_x \log p_{\text{data}}(x)$: use Langevin MCMC to sample.
- Let's learn the score function from data — **score matching** (SM) (Hyvärinen 2005).

Explicit Score Matching:

$$\operatorname{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \| s_\theta(X_i) - \underbrace{\nabla \log p_{\text{data}}(X_i)}_{\text{intractable}} \|^2.$$

Fortunately, Hyvärinen 2005 shows that the above is equivalent to

Implicit Score Matching (ISM):

$$\operatorname{argmin}_{\theta \in \Theta} \sum_{i=1}^{n} \left(\operatorname{Trace}(\nabla_x s_\theta(X_i)) + \frac{1}{2} \| s_\theta(X_i) \|^2 \right).$$

ESM vs ISM

Why?

$$\begin{aligned}
\mathbb{E}\,\mathsf{ESM}(\theta) &= \int \|s_\theta(x) - \nabla \log p_{\mathsf{data}}(x)\|^2 p_{\mathsf{data}}(\mathrm{d}x) \\
&= \int \|s_\theta(x)\|^2 p_{\mathsf{data}}(\mathrm{d}x) - \int 2 s_\theta(x)^\top \frac{\nabla p_{\mathsf{data}}(x)}{p_{\mathsf{data}}(x)} p_{\mathsf{data}}(x)\mathrm{d}x + \text{constant in } \theta \\
&\overset{\circ}{=} \int \|s_\theta(x)\|^2 p_{\mathsf{data}}(\mathrm{d}x) - 2\int s_\theta(x)^\top \nabla p_{\mathsf{data}}(x)\mathrm{d}x
\end{aligned}$$

integration by parts, assuming $\lim_{\|x\|\to\infty} p_{\mathsf{data}}(x) s_\theta(x) = 0$

$$\begin{aligned}
&\overset{\circ}{=} \int \|s_\theta(x)\|^2 p_{\mathsf{data}}(\mathrm{d}x) - 2\underbrace{\left[s_\theta^\top \nabla p_{\mathsf{data}}(x)\right]_{-\infty}^{\infty}}_{=0} + 2\int \left[p_{\mathsf{data}}(x)\nabla \cdot s_\theta(x)\right] \\
&\overset{\circ}{=} \int \left[\|s_\theta(x)\|^2 + 2\mathsf{Trace}(s_\theta(x))\right] p_{\mathsf{data}}(\mathrm{d}x).
\end{aligned}$$

Ingredients 2: (de)Noising

- Learning $\nabla \log p_{\text{data}}$ is hard if p_{data} is supported on a low-dimensional manifold (e.g. images)
- Estimation hard away from modes.
- Even if we could learn it, sampling is slow (Langevin MCMC), as p_{data} typically multimodal or complex geometry.
- **Idea:** regularise by adding **noise**!

Ingredients 2: (de)Noising

- Add Gaussian noise to p_{data}: $p_\sigma = p_{\text{data}} * \mathcal{N}(0, \sigma^2 I)$
- p_σ has full support, smoother, easier to learn
- **Denoising Score Matching (DSM)** Vincent 2011: learn $\nabla \log p_\sigma$ using score matching.
- Data? Easy!

$$Y_i = X_i + \sigma Z_i, \quad Z_i \sim \mathcal{N}(0, I), \quad i = 1, \ldots, n.$$

- **Score Matching objective applied to p_σ:**

$$\mathsf{ESM}_\sigma(\theta) = \mathsf{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \| s_\theta(Y_i) - \underbrace{\nabla \log p_\sigma(Y_i)}_{\text{intractable}} \|^2.$$

- Yet again intractable.

Ingredients 2: (de)Noising

Here comes the magic:

$$
\begin{aligned}
\mathbb{E}\,\mathsf{ESM}_\sigma(\theta) &= \int \|s_\theta(y) - \nabla \log p_\sigma(y)\|^2 p_\sigma(\mathrm{d}y) \\
&\overset{\circ}{=} \int \|s_\theta(y)\|^2 p_\sigma(\mathrm{d}y) - 2\int s_\theta(y)^\top \nabla p_\sigma(y)\mathrm{d}y
\end{aligned}
$$

Write $q_\sigma(y|x) := \mathcal{N}(x, \sigma^2 I)$, the density of $Y|X = x$. Then

$$
\begin{aligned}
\nabla p_\sigma(y) &= \nabla_y \int p_{\mathsf{data}}(x) q_\sigma(y|x)\mathrm{d}x \\
&= \int p_{\mathsf{data}}(x) \nabla_y q_\sigma(y|x)\mathrm{d}x \\
&= \int p_{\mathsf{data}}(x) \frac{\nabla_y q_\sigma(y|x)}{q_\sigma(y|x)} q_\sigma(y|x)\mathrm{d}x \\
&= \int p_{\mathsf{data}}(x) \nabla_y \log q_\sigma(y|x) q_\sigma(y|x)\mathrm{d}x.
\end{aligned}
$$

Thus

$$
\begin{aligned}
\mathbb{E}\,\mathsf{ESM}_\sigma(\theta) &\overset{\circ}{=} \int \|s_\theta(y)\|^2 p_\sigma(\mathrm{d}y) - 2\int s_\theta(y)^\top \nabla p_\sigma(y)\mathrm{d}y \\
&\overset{\circ}{=} \int\int \|s_\theta(y)\|^2 p_\sigma(\mathrm{d}y) - 2\int_x\int_y s_\theta(y)^\top \nabla_y \log q_\sigma(y|x) q_\sigma(y|x)\mathrm{d}y\, p_{\mathsf{data}}(x)\mathrm{d}x \\
&\overset{\circ}{=} \iint \|s_\theta(y) - \nabla_y \log q_\sigma(y|x)\|^2 q_\sigma(y|x)\mathrm{d}y\, p_{\mathsf{data}}(x)\mathrm{d}x.
\end{aligned}
$$

The above is fully tractable and suggests using the following objective:

Denoising Score Matching (DSM): $\quad \mathrm{argmin}_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n} \|s_\theta(Y_i) - \nabla_y \log q_\sigma(Y_i|X_i)\|^2.$

where $X_i \sim p_{\mathsf{data}}$, $Y_i|X_i \sim \mathcal{N}(X_i, \sigma^2 I)$.

Denoising score matching

So does it work?

Benefits: much more stable than ISM.

Drawbacks: choice of σ critical

- σ too small: $p_\sigma \approx p_{\mathsf{data}}$, problems of ISM remain.
- σ too large: denoising too hard, $\nabla \log p_\sigma$ very different from $\nabla \log p_{\mathsf{data}}$.

Beginnings of Diffusion Models

Solution: Y. Song and Ermon 2019 suggest using **multiple noise** levels $\sigma_1 < \sigma_2 < \cdots < \sigma_K$, and learning $s_\theta(x, \sigma)$ using a **noise-conditional score network**.

This creates a sequence of auxiliary targets:

$$p_{\text{data}}^{\sigma_1}, p_{\text{data}}^{\sigma_2}, \ldots, p_{\text{data}}^{\sigma_K}.$$

Small σ: bad score estimation, hard to sample close to p_{data}.

Large σ: good score estimation, easy to sample, far from p_{data}.

Use a **noise-dependent** neural network to learn

$$s_\theta(x, \sigma) \approx (\sigma, x) \mapsto \nabla \log p_{\text{data}}^{\sigma}(x).$$

Finally use **annealed Langevin dynamics** to guide samples through

$$p_{\text{data}}^{\sigma_K} \to p_{\text{data}}^{\sigma_{K-1}} \to \cdots \to p_{\text{data}}^{\sigma_1} \approx p_{\text{data}}.$$

Denoising diffusion models: Discrete time

Ho, Jain, and Abbeel 2020 propose the following discrete time approach:

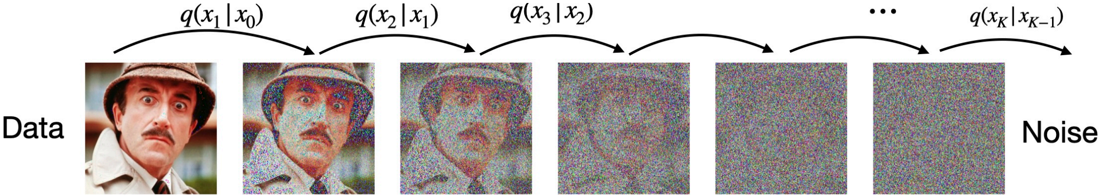
- **Forward noising process:**

$$X_k = \sqrt{1-\beta_k} X_{k-1} + \sqrt{\beta_k} Z_k, \quad Z_k \sim \mathcal{N}(0, I), \quad k = 1, \dots, K.$$

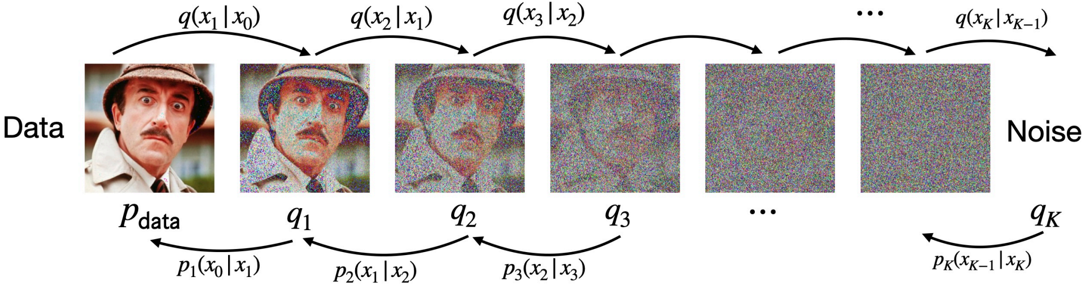
- Equivalently, this is a **discrete time Markov process** $X_0, \dots, X_K$ with

$$X_0 \sim p_{\mathsf{data}}, \qquad q_j(X_j | X_{j-1}) = \mathcal{N}(\sqrt{\alpha_j} X_{j-1}, \beta_j I), \qquad \alpha_j := 1 - \beta_j.$$

- Notice that $X_k \mid X_0 \sim \mathcal{N}\big(\sqrt{\bar{\alpha}_k}\, X_0,\ (1-\bar{\alpha}_k)\, I\big)$, where $\bar{\alpha}_k := \prod_{i=1}^k \alpha_i$ and $\bar{\alpha}_0 := 1$.



Denoising diffusion models: Discrete time



If $\bar{\alpha}_K \approx 0$, then $X_K \approx \mathcal{N}(0, I)$.

Generative model: Sample

$$Y_K \sim \mathcal{N}(0, I), \qquad Y_{k-1}|Y_k \sim p_k(Y_{k-1}|Y_k), \quad k = K, K-1, \ldots, 1.$$

Then $Y_0 \approx p_{\text{data}}$.

DDM in discrete time: Denoising I

Problem: We don't know the **reverse denoising process** $p_k(X_{k-1}|X_k)$, $k = 1, \dots, K$.

Simply using Bayes' rule we have that

$$p_k(x_{k-1}|x_k) = \frac{q_{k-1}(x_{k-1})q_k(x_k|x_{k-1})}{q_k(x_k)}, \qquad q_k = \mathrm{Law}(X_k).$$

Problem: q_k is intractable.

Solution: Taylor expand

$$\log q_{k-1}(x_{k-1}) \approx \log q_k(x_{k-1}) \approx q_k(x_k) - (x_k - x_{k-1})^\top \nabla \log q_k(x_k) + \dots$$

Ignoring the higher order terms and plugging in we get

$$p_k(x_{k-1}|x_k) \approx \frac{1}{\sqrt{2\pi\beta_k}^d} \exp\left(-\frac{1}{2\beta_k^2}\|x_k - \sqrt{\alpha_k}x_{k-1}\|^2 - (x_k - x_{k-1})^\top \nabla \log q_k(x_k)\right).$$

DDM in discrete time: Denoising II

Completing the square we get

$$
\begin{aligned}
&\propto \frac{1}{\sqrt{2\pi\beta_k}^d} \exp\left[-\frac{1}{2\beta_k^2}\left\{\alpha_k\|x_{k-1}\|^2 - 2x_{k-1}^\top\left(\sqrt{\alpha_k}x_k + \beta_k\nabla\log q_k(x_k)\right)\right\}\right] \\
&\propto \frac{1}{\sqrt{2\pi\beta_k}^d} \exp\left[-\frac{\alpha_k}{2\beta_k^2}\left\{\|x_{k-1}\|^2 - 2x_{k-1}^\top\left(\frac{x_k}{\sqrt{\alpha_k}} + \frac{\beta_k}{\sqrt{\alpha_k}}\nabla\log q_k(x_k)\right)\right\}\right] \\
&\propto \frac{1}{\sqrt{2\pi\beta_k}^d} \exp\left[-\frac{\alpha_k}{2\beta_k^2}\left\|x_k - \frac{1}{\sqrt{\alpha_k}}(x_k + \beta_k\nabla\log q_k(x_k))\right\|^2\right],
\end{aligned}
$$

and thus

$$
X_{k-1}|X_k = x_k \approx \mathcal{N}\left(\frac{1}{\sqrt{\alpha_k}}\left(x_k + \beta_k\nabla\log q_k(x_k)\right), \frac{\beta_k^2}{\alpha_k}I\right).
$$

BUT: we don't know $\nabla\log q_k$.
Recall: q_k is the law of the noisy sample X_k.

DDM in discrete time: Denoising I

Alternative derivation: We can explicitly write down $q(x_{k-1}|x_k, x_0)$ as

$$q(x_{k-1}|x_0, x_k) = \mathcal{N}(x_{k-1}; \tilde{\mu}_k(x_0, x_k), \tilde{\beta}_k I), \qquad \text{where}$$
$$\tilde{\mu}_k(x_0, x_k) = \frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k} x_k + \frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k} x_0, \qquad \tilde{\beta}_k = \frac{(1-\bar{\alpha}_{k-1})\beta_k}{1-\bar{\alpha}_k}.$$

Thus

$$\begin{aligned} q(x_{k-1}|x_k) &= \int q(x_{k-1}|x_0, x_k) p(x_0|x_k) \mathrm{d}x_0 \\ &= \int \mathcal{N}(x_{k-1}; \tilde{\mu}_k(x_0, x_k), \tilde{\beta}_k I) p(x_0|x_k) \mathrm{d}x_0 \end{aligned}$$

and approximating $p(x_0|x_k)$ by a delta mass at its mean $\mathbb{E}[x_0|x_k]$ we get

$$\approx \mathcal{N}\left(x_{k-1}; \tilde{\mu}_k(x_k, \mathbb{E}[x_0|x_k]), \tilde{\beta}_k I\right).$$

DDM in discrete time: Denoising II

This approximation is reasonable if k is not too large and the discretisation step is quite fine. Continuing

$$x_{k-1}|x_k \approx \mathcal{N}\left(\frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}x_k + \frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\,\mathbb{E}[x_0|x_k], \frac{(1-\bar{\alpha}_{k-1})\beta_k}{1-\bar{\alpha}_k}I\right).$$

This simplifies to

$$x_{k-1}|x_k \approx \mathcal{N}\left(\frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k}x_k + \frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\,\mathbb{E}[x_0|x_k], \frac{(1-\bar{\alpha}_{k-1})\beta_k}{1-\bar{\alpha}_k}I\right).$$

Problem: the unknown expression now is a conditional expectation $\mathbb{E}[X_0|X_k = x_k]$.
Idea: learn it using Denoising Score Matching!

DDM in discrete time: Tweedie's formula I

We've found two distinct approximations for $q(x_{k-1}|x_k)$:

$$
\begin{aligned}
p_k(x_{k-1}|x_k) &\approx \mathcal{N}\left(\frac{1}{\sqrt{\alpha_k}}\left(x_k + \beta_k \nabla \log q_k(x_k)\right), \frac{\beta_k^2}{\alpha_k} I\right), \\
&\approx \mathcal{N}\left(\frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k} x_k + \frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k}\, \mathbb{E}[X_0|X_k = x_k], \frac{(1-\bar{\alpha}_{k-1})\beta_k}{1-\bar{\alpha}_k} I\right).
\end{aligned}
$$

DDM in discrete time: Tweedie's formula

Question: How are they related?

The answer is given by **Tweedie's formula** (Efron 2011):

Theorem

Let $X_0 \sim p$ *and* $Y|X_0 = x \sim \mathcal{N}(x, \sigma^2 I)$. *Then*

$$\mathbb{E}[X_0|Y = y] = y + \sigma^2 \nabla \log p_\sigma(y), \qquad p_\sigma = p * \mathcal{N}(0, \sigma^2 I).$$

Proof.

The proof is simple and goes as follows:

$$\nabla \log p_\sigma(y) = \frac{\nabla p_\sigma(y)}{p_\sigma(y)} = \int \frac{y - x}{\sigma^2} \frac{1}{(2\pi\sigma^2)^{d/2}} \frac{p(x) e^{-\frac{\|y-x\|^2}{2\sigma^2}}}{p_\sigma(y)} \mathrm{d}x = \frac{1}{\sigma^2} \mathbb{E}[Y - X_0|Y = y],$$

where $X_0 \sim p_{\text{data}}, Y|X_0 = x \sim \mathcal{N}(x, \sigma^2 I)$. Rearranging gives the result. □

Tweedie's formula in our case I

By a simple modification of the proof of Tweedie's formula we get that

$$
\begin{aligned}
\tilde{p}_{a,b} &= \mathsf{Law}(aX_0 + bZ), \quad X_0 \sim p_{\mathsf{data}},\ Z \sim \mathcal{N}(0, I), \\
\nabla \log p_{a,b}(y) &= \frac{1}{b}\,\mathbb{E}[Y - aX_0 | Y = y] \\
\implies \mathbb{E}[X_0 | Y = y] &= \frac{1}{a}\left(y - b\nabla \log p_{a,b}(y)\right).
\end{aligned}
$$

Therefore in our case, since $X_k | X_0 \sim \mathcal{N}(\sqrt{\bar{\alpha}_k} X_0, (1 - \bar{\alpha}_k) I)$, we have

$$
\mathbb{E}[X_0 | X_k = x_k] = \frac{1}{\sqrt{\bar{\alpha}_k}}\left(x_k + (1 - \bar{\alpha}_k)\nabla \log q_k(x_k)\right).
$$

Plugging in to the formula for $\tilde{\mu}_k$ we get

$$
\tilde{\mu}_k(x_k, \mathbb{E}[X_0 | X_k = x_k]) = \frac{\sqrt{\alpha_k}(1 - \bar{\alpha}_{k-1})}{1 - \bar{\alpha}_k} x_k + \frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1 - \bar{\alpha}_k}\,\mathbb{E}[X_0 | X_k = x_k]
$$

Tweedie's formula in our case II

$$
\begin{aligned}
&= \frac{\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})}{1-\bar{\alpha}_k} x_k + \frac{\sqrt{\bar{\alpha}_{k-1}}\beta_k}{1-\bar{\alpha}_k} \frac{1}{\sqrt{\bar{\alpha}_k}} \left(x_k + (1-\bar{\alpha}_k)\nabla \log q_k(x_k)\right) \\
&= \frac{1}{1-\bar{\alpha}_k}\left[\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1})x_k + \frac{\beta_k}{\sqrt{\alpha_k}} x_k\right] + \frac{\beta_k}{\sqrt{\alpha_k}} \nabla \log q_k(x_k).
\end{aligned}
$$

Using $\beta_k = 1 - \alpha_k$ and $\bar{\alpha}_k = \bar{\alpha}_{k-1}\alpha_k$ we can combine the x_k terms to

$$
\sqrt{\alpha_k}(1-\bar{\alpha}_{k-1}) + \frac{\beta_k}{\sqrt{\alpha_k}} = \frac{\alpha_k(1-\bar{\alpha}_{k-1})}{\sqrt{\alpha_k}} + \frac{1-\alpha_k}{\sqrt{\alpha_k}} = \frac{\alpha_k - \alpha_k\bar{\alpha}_{k-1} + 1 - \alpha_k}{\sqrt{\alpha_k}} = \frac{1-\bar{\alpha}_k}{\sqrt{\alpha_k}}.
$$

Thus overall we have

$$
\begin{aligned}
\tilde{\mu}_k(x_k, \mathbb{E}[X_0|X_k = x_k]) &= \frac{1-\bar{\alpha}_k}{\sqrt{\alpha_k}} x_k + \frac{\beta_k}{\sqrt{\alpha_k}} \nabla \log q_k(x_k) \\
&= \boxed{\frac{1}{\sqrt{\alpha_k}}\left(x_k + \beta_k \nabla \log q_k(x_k)\right)}.
\end{aligned}
$$

Parameterisation and learning

Recall that we have shown that

$$X_{t-1}|X_t = x_t \sim p_t(\cdot|x_t) \approx \mathcal{N}\left(\frac{1}{\sqrt{\alpha_t}}\left(x_t + \beta_t \nabla \log q_t(x_t)\right), \frac{\beta_t^2}{\alpha_t} I\right).$$

Problem: we don't know $\nabla \log q_t$.
Solution: Let's replace $\nabla \log q_t(x_t)$ by a neural network approximation $s_\theta : [0, T] \times \mathbb{R}^d \to \mathbb{R}^d$ and define

$$p_t^\theta(x_{t-1}|x_t) = \mathcal{N}\left(s_\theta(t, x_t), \frac{\beta_t^2}{\alpha_t} I\right).$$

Here we think of t as a **continuous time variable**, and s_θ as a time-dependent neural network.

How do we train s_θ?

Parameterisation and learning (ctd.)

Key observation: Let's do regression!
If we could solve

$$\arg\min_{\theta\in\Theta} \mathbb{E}_{X_t\sim q_t}\left[\sum_{t=1}^{T} \|\nabla \log q_t(X_t) - s_\theta(t, X_t)\|^2\right], \qquad \text{(ESM)} \quad \{\text{ec}$$

we would be done!
Problems:

(1) We don't know $\nabla \log q_t$.

(2) The expectation is over the unknown distribution of the forward noising process q.

Second problem easy! Sample from q_t by sampling $X_0 \sim p_{\text{data}}$ and then running the forward noising process.
The first problem is more serious.

Parameterisation and learning (ctd.)

Recall the equivalent **Denoising Score Matching (DSM):**

$$\arg\min_{\theta\in\Theta} \sum_{t=1}^{T} \iint \left[\|\nabla \log q_{t|0}(x_t|x_0) - s_\theta(t, x_t)\|^2\right] p_{\mathsf{data}}(\mathrm{d}x_0) q_{t|0}(x_t|x_0) \tag{DSM}$$ {ec

where as we saw earlier it is easy to show that $q_{t|0}(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar\alpha_t}x_0, (1-\bar\alpha_t)I)$.

In practice: empirical DSM objective of the form

$$\arg\min_{\theta\in\Theta} \sum_{t=1}^{T} \sum_{i=1}^{N} \left[\|\nabla \log q_{t|0}(X_t^{(i)}|X_0^{(i)}) - s_\theta(t, X_t^{(i)})\|^2\right], \tag{emp-DSM}$$ {ec

$$X_0^{(i)} \overset{i.i.d.}{\sim} p_{\mathsf{data}},\ X_t^{(i)}|X_0^{(i)} \sim q_{t|0}(\cdot|X_0^{(i)}). \tag{1}$$

In fact the sum over t can be replaced by an expectation over the uniform distribution on $[0, T]$.

Key point: It is very important for the efficiency of training that we can sample (X_0, X_t) easily, without having to use s_θ or sample intermediate steps of the forward process.

DDM in discrete time: ELBO objective

Ho, Jain, and Abbeel 2020 start deriving their objective by viewing the sampling distribution $p^\theta \in \mathcal{P}(\mathbb{R}^d)$ of a DDM as a **latent variable model**

$$p^\theta(x_0) = \int p^\theta(x_0, \ldots, x_T) \mathrm{d}x_{1:T},$$

where $x_1, \ldots, x_T \in \mathbb{R}^d$ are the intermediate steps of the forward process.

Different parameterisation for the reverse denoising kernels:

$$p^\theta_{t|t+1}(x_t | x_{t+1}) = \mathcal{N}(x_t; \mu_\theta(t+1, x_{t+1}), \sigma^2_{t+1} \mathbb{1}), \qquad t = 0, \ldots, T-1 \tag{2}$$

$$p^\theta(x_{0:T}) = p_T(x_T) p^\theta_{T-1|T}(x_{T-1}|x_T) \ldots p^\theta_{0|1}(x_0|x_1), \tag{3}$$

where $p_T(x_T) = \mathcal{N}(x_T; 0, \mathbb{1})$.

DDM in discrete time: ELBO objective (ctd.)

ELBO: train by maximising average log-likelihood of data

$$\begin{aligned}\mathbb{E}_{p_{\mathsf{data}}}[\log p^\theta(x_0)] &= \mathbb{E}_{x_0\sim p_{\mathsf{data}}}\left[\log \int p^\theta(x_{0:T})\mathrm{d}x_{1:T}\right] \\ &= \mathbb{E}_{x_0\sim p_{\mathsf{data}}}\left[\log \int \frac{p^\theta(x_{0:T})}{q_{1:T|0}(x_{1:T}|x_0)} q_{1:T|0}(x_{1:T}|x_0)\mathrm{d}x_{1:T}\right] \\ &\geqslant \mathbb{E}_{x_{0:T}\sim q}\left[\log \frac{p^\theta(x_{0:T})}{q_{1:T|0}(x_{1:T}|x_0)}\right].\end{aligned}$$

where recall $q \in \mathcal{P}(\mathbb{R}^{d\times T})$ **is the joint distribution of the forwards process**.

Remark: notice that in contrast to typical variational inference, q is fixed and known, and we are only optimising over p^θ.

DDM in discrete time: ELBO objective (ctd.)

ELBO can be re-arranged into terms involving only consecutive pairs of variables

$$\mathbb{E}_{p_{\text{data}}}[\log p^\theta(x_0)] \geqslant \mathbb{E}_{x_{0:T}\sim q}\left[\log \frac{p^\theta(x_{0:T})}{q_{1:T|0}(x_{1:T}|x_0)}\right]$$
$$= \mathbb{E}_{x_{0:T}\sim q}\left[\log p_T(x_T) + \sum_{t=2}^{T} \log \frac{p^\theta_{t-1|t}(x_{t-1}|x_t)}{\color{red}{q_{t|t-1}(x_t|x_{t-1})}} + \log \frac{p^\theta_{0|1}(x_0|x_1)}{q_{1|0}(x_1|x_0)}\right].$$

Recall that

$$\begin{aligned}
\color{red}{q_{t|t-1}(x_t|x_{t-1})} &= q_{t|t-1,0}(x_t|x_{t-1}, x_0) \qquad \text{(by Markov property)} \\
&= \frac{q_{t,t-1|0}(x_t, x_{t-1}|x_0)}{q_{t-1|0}(x_{t-1}|x_0)} \\
&= \frac{q_{t-1|t,0}(x_{t-1}|x_t, x_0)q_{t|0}(x_t|x_0)}{q_{t-1|0}(x_{t-1}|x_0)}.
\end{aligned}$$

DDM in discrete time: ELBO objective (ctd.)

Plugging in the expression for $q_{t|t-1}$ we get

$$
\begin{aligned}
&\mathbb{E}_{p_{\mathsf{data}}}[\log p^{\theta}(x_0)] \\
&\geqslant \mathbb{E}_{x_{0:T}\sim q}\left[\log p_T(x_T) + \sum_{t=2}^{T}\log \frac{p^{\theta}_{t-1|t}(x_{t-1}|x_t)}{q_{t-1|t,0}(x_{t-1}|x_t,x_0)}\frac{q_{t-1|0}(x_{t-1}|x_0)}{q_{t|0}(x_t|x_0)} + \log\frac{p^{\theta}_{0|1}(x_0|x_1)}{q_{1|0}(x_1|x_0)}\right] \\
&= \mathbb{E}_{x_{0:T}\sim q}\left[\log p_T(x_T) + \sum_{t=2}^{T}\log \frac{p^{\theta}_{t-1|t}(x_{t-1}|x_t)}{q_{t-1|t,0}(x_{t-1}|x_t,x_0)} + \log\frac{p^{\theta}_{0|1}(x_0|x_1)}{q_{1|0}(x_1|x_0)} + \sum_{t=2}^{T}\log\frac{q_{t-1|0}(x_{t-1}|x_0)}{q_{t|0}(x_t|x_0)}\right. \\
&= \mathbb{E}_{x_{0:T}\sim q}\left[\log p_T(x_T) + \sum_{t=2}^{T}\log \frac{p^{\theta}_{t-1|t}(x_{t-1}|x_t)}{q_{t-1|t,0}(x_{t-1}|x_t,x_0)} + \log\frac{p^{\theta}_{0|1}(x_0|x_1)}{q_{1|0}(x_1|x_0)}\right. \\
&\qquad\qquad\qquad \left. + \log q_{1|0}(x_1|x_0) - \log q_{T|0}(x_T|x_0)\right]
\end{aligned}
$$

DDM in discrete time: ELBO objective (ctd.)

Telescoping the final sum

$$= \mathbb{E}_{x_{0:T}\sim q}\left[\log\frac{p_T(x_T)}{q_{T|0}(x_T|x_0)} + \sum_{t=2}^{T}\log\frac{p^\theta_{t-1|t}(x_{t-1}|x_t)}{q_{t-1|t,0}(x_{t-1}|x_t,x_0)} + \log p^\theta_{0|1}(x_0|x_1)\right]$$

$$= \mathsf{E}_{x_{0:T}\sim q}\left[-\underbrace{\mathsf{KL}\left(q_{T|0}(\cdot|x_0)\middle\| p_T(\cdot)\right)}_{L_T} - \sum_{t=2}^{T}\underbrace{\mathsf{KL}\left(q_{t-1|t,0}(\cdot|x_t,x_0)\middle\| p^\theta_{t-1|t}(\cdot|x_t)\right)}_{L_{t-1}}\right.$$

$$\left. - \underbrace{\log p^\theta_{0|1}(x_0|x_1)}_{L_0}\right].$$

Recall:

(1) $q_{t-1|t,0}(x_{t-1}|x_t,x_0) = \mathcal{N}(x_{t-1};\tilde{\mu}_t(x_t,x_0),\tilde{\beta}t\mathbb{1})$, where

$$\tilde{\mu}_t(x_t,x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{1-\beta_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t, \qquad \tilde{\beta}_t = \frac{(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\beta_t.$$

DDM in discrete time: ELBO objective (ctd.)

(2) The backwards kernels are parameterised as

$$p^{\theta}_{t-1|t}(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(t, x_t), \sigma_t^2 \mathbb{1}), \qquad t = 1, \ldots, T.$$

(3)or $\mu_i \in \mathbb{R}^d$, and Σ_i d-dimensional covariance matrices,

$$\begin{aligned} &\mathsf{KL}\left(\mathcal{N}(\mu_1, \Sigma_1) \middle\| \mathcal{N}(\mu_2, \Sigma_2)\right) \\ &= \frac{1}{2}\left[\log \frac{\det(\Sigma_2)}{\det(\Sigma_1)} + \frac{1}{2}\mathrm{trace}(\Sigma_2^{-1}\Sigma_1) + (\mu_2 - \mu_1)^{\intercal}\Sigma_2^{-1}(\mu_2 - \mu_1) - d\right]. \end{aligned} \tag{4}$$

{ec

Therefore we can now compute

$$L_{t-1} = \mathsf{KL}\left(q_{t-1|t,0}(\cdot|x_t, x_0) \middle\| p^{\theta}_{t-1|t}(\cdot|x_t)\right).$$

DDM in discrete time: ELBO objective (ctd.)

Using (4) with $\mu_1 = \tilde{\mu}_t(x_t, x_0), \Sigma_1 = \tilde{\beta}_t \mathbb{1}, \mu_2 = \mu_\theta(t, x_t), \Sigma_2 = \sigma_t^2 \mathbb{1}$ we get

$$\begin{aligned} L_{t-1} &= \frac{1}{2}\left[d \log \frac{\sigma_t^2}{\tilde{\beta}_t} + \frac{d}{2}\frac{\tilde{\beta}_t}{\sigma_t^2} + \frac{1}{\sigma_t^2}\left\| \mu_\theta(t, x_t) - \tilde{\mu}_t(x_t, x_0) \right\|^2 - d \right] \\ &\overset{\circ}{=} \frac{1}{2\sigma_t^2}\mathbb{E}_q \left\| \mu_\theta(t, x_t) - \tilde{\mu}_t(x_t, x_0) \right\|^2 . \end{aligned}$$

The term then fits a neural network taking as input t, x_t to predict $\tilde{\mu}_t(x_t, x_0)$.

When $x_0, x_t \sim q_{0t}$ we can also reparameterise x_t as

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\xi, \qquad \xi \sim \mathcal{N}(0, \mathbb{1}),$$

and therefore

$$\tilde{\mu}_t(x_t, x_0) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}}\xi \right).$$

DDM in discrete time: ELBO objective (ctd.)

At this point, since we want μ_t^θ to take x_t as input, we may reparameterise it as

$$\mu^\theta(t, x_t) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \xi^\theta(t, x_t) \right). \tag{5}$$

With this parameterisation L_{t-1} becomes

$$\begin{aligned} L_{t-1} &= \frac{1}{2\sigma_t^2} \mathbb{E}_{x_0 \sim p_{\mathsf{data}}, \xi \sim \mathcal{N}} \left\| \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \xi^\theta(t, x_t) \right) - \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \xi \right) \right\|^2 + \text{const.} \\ &= \frac{\beta_t^2}{2\alpha_t(1-\bar{\alpha}_t)\sigma_t^2} \mathsf{E}_{x_0 \sim p_{\mathsf{data}}, \xi \sim \mathcal{N}} \left\| \xi^\theta \left(t, \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t} \xi \right) - \xi \right\|^2 + \text{const.}. \end{aligned}$$

Finally we have the L_0 which is simply given by

$$L_0 = \mathbb{E}_q[\log p^\theta_{0|1}(x_0|x_1)] = \mathbb{E}_q[-\frac{1}{2\sigma_1^2} \|x_0 - \mu_\theta(1, x_1)\|^2] + \text{const.}.$$

DDM in discrete time: ELBO objective (ctd.)

Notice that since $x_1 = \sqrt{1-\beta_1}x_0 + \beta_1\xi$, we we can write $\boxed{x_0 = \frac{1}{\sqrt{1-\beta_1}}(x_1 - \beta_1\xi)}$.

Reparameterise $\mu_\theta(1, x_1)$ as $\boxed{\mu_\theta(1, x_1) = \frac{1}{\sqrt{1-\beta_1}}(x_1 - \beta_1\xi^\theta(1, x_1))}$.

Thus

$$
\begin{aligned}
L_0 &= \frac{1}{2\sigma_1^2}\,\mathbb{E}_{x_0,\xi}\left\|\frac{1}{\sqrt{1-\beta_1}}\Big[x_1(x_0,\xi) - \beta_1\xi\Big] - \frac{1}{\sqrt{1-\beta_1}}\Big[x_1(x_0,\xi) - \beta_1\xi^\theta(1, x_1(x_0,\xi))\Big]\right\|^2 \\
&= \frac{\beta_1^2}{2(1-\beta_1)\sigma_1^2}\,\mathbb{E}_{x_0,\xi}\left\|\xi - \xi^\theta\Big(1, \sqrt{1-\beta_1}x_0 + \beta_1\xi\Big)\right\|^2.
\end{aligned}
$$

DDM in discrete time: ELBO objective (ctd.)

Putting everything together we obtain the objective

$$\mathcal{L}_{\mathsf{ELBO}}(\theta) = \sum_{t=0}^{T-1} \frac{\beta_t^2}{2\alpha_t(1-\tilde{\alpha}_t)\sigma_t^2} \mathbb{E}_{x_0\sim p_{\mathsf{data}},\xi\sim\mathcal{N}} \left\| \xi^\theta\left(t, \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\xi\right) - \xi \right\|^2, \tag{6}$$

where $\tilde{\alpha}_0 = 1$ and $\tilde{\alpha}_t = \bar{\alpha}_t$ for $t > 1$. Ho, Jain, and Abbeel 2020 suggest choosing either $\sigma_t^2 := \beta_t$ or $\sigma_t^2 = \beta_t(1-\bar{\alpha}_{t-1})/(1-\bar{\alpha}_t)$.

Remark

Ho, Jain, and Abbeel 2020 report that they had good results with a simplified object given by

Simple DDPM objective

$$\mathcal{L}_{\mathsf{simple}}(\theta) := \mathbb{E}_{\substack{x_0\sim p_{\mathsf{data}} \\ \xi\sim\mathcal{N}(0,1) \\ t\sim U(0:T)}} \left\| \xi^\theta\left(t, \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\xi\right) - \xi \right\|^2 \tag{7}$$

{eq:si

where one replaces the sum over t with sampling a random time uniformly from $[0:T]$.

DDM in discrete time: the generative model

Given a trained model $\theta^\star$, we can sample from $p^{\theta^\star}$ by

(1) Sample $X_T \sim \mathcal{N}(0, I)$

(2) For $t = T, T-1, \ldots, 1$ sample

$$X_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(X_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\xi_{\theta^\star}(t, X_t)\right) + \sigma_t Z_t, \qquad Z_t \sim \mathcal{N}(0, I).$$

Noising schedules

- The choice of the noising schedule $(\beta_t)_{t=1}^T$ is very important.
- Ho, Jain, and Abbeel 2020 suggest a linear schedule $\beta_t = \beta_{\min} + \frac{t-1}{T-1}(\beta_{\max} - \beta_{\min})$ with $\beta_{\min} = 0.0001$ and $\beta_{\max} = 0.02$.
- There are other choices, like quadratic or cosine schedules A. Q. Nichol and Dhariwal 2021.
- These attempt to allocate more time points towards the early stages of noising, where the data distribution is more complex.
- This has the effect of making the learning of the target more gradual.
- Also most schedules tend to stop just before $\beta_T = 0$ where the score blows up.

Continuous time diffusion

Continuous time DDMs

Soon after Ho, Jain, and Abbeel 2020, another formulation appeared in Y. Song, Sohl-Dickstein, et al. 2021.

This is based on stochastic differential equations and is very elegant.

Again suppose that we have samples $X_i \sim p_{\text{data}}$.

Noising process: given by the **Ornstein-Uhlenbeck SDE**

$$\mathrm{d}Y_t = -Y_t \mathrm{d}t + \sqrt{2}\mathrm{d}W_t, \qquad Y_0 \sim p_{\text{data}} \tag{8}$$

{ec

where W_t is a Brownian motion.

Notation: Write

- $q_{t|s}(x_t|x_s)$ for the transition of the forward process
- p_t for the distribution of Y_t.

The latent variable here is a continuous time path $(Y_t : t \geqslant 0)$ instead of a vector $(X_1, \ldots, X_T)$.

Continuous time DDMs (ctd.)

This is an auto-regressive process, the continuous time equivalent of an AR(1) process and has several nice properties.

(a) $Y_{t+h}|Y_t \sim \mathcal{N}(\mathrm{e}^{-h}Y_t, (1-\mathrm{e}^{-2h})I)$, for $t, h \geqslant 0$.

(b) It is a Markov process, with $\mathcal{N}(0, I)$ as its unique stationary distribution.

(c) It forgets its initialisation at an **dimension-free, exponential rate**:

$$Y_T \approx \mathcal{N}(0, I), \qquad T \gg 1.$$

Continuous time DDMs—Time Reversal

The real magic happens when we try to reverse the process.

Theorem (Anderson 1982)

Let Y_t be the solution to the SDE (8) with $Y_0 \sim p_{\mathsf{data}}$.

*Then for any $T > 0$, the **time reversal** $Z_t := Y_{T-t}$, $t \in [0, T]$ is the solution to the SDE*

$$\mathrm{d}Z_t = -[Z_t + 2\nabla \log p_{T-t}(Z_t)]\mathrm{d}t + \sqrt{2}\mathrm{d}B_t, \qquad Z_T \sim \mathcal{N}(0, I). \tag{9}$$

Remark

Notice that in this formulation the score appears naturally.

Conditional diffusion models and guidance

Conditional score matching

Goal: learn to sample from Law$(X|Y=y)$ for some y,

given: samples $(X_i, Y_i) \sim p_{\mathsf{data}}$ from the joint.

If: samples $X_1, \ldots, X_n$ from conditional Law$(X|Y=y)$ previous techniques would work.

Let's see what we can do with samples from the joint.

Maximizing the standard ESM objective on the joint:

$$\arg\min_{\theta \in \Theta} \mathbb{E}\left[\|\nabla_x \log p_{\mathsf{data}}(X,Y) - s_\theta(X,Y)\|^2\right]. \tag{10}$$

Conditional score matching

Notice that

$$
\begin{aligned}
&\iint \|\nabla_x \log p_{\mathsf{data}}(x,y) - s_\theta(x,y)\|^2 p_{\mathsf{data}}(x,y)\mathrm{d}x\mathrm{d}y \\
&\overset{\circ}{=} \iint \left[\|s_\theta(x,y)\|^2 - 2\nabla_x \log p_{\mathsf{data}}(x,y)^\top s_\theta(x,y)\right] p_{\mathsf{data}}(x,y)\mathrm{d}x\mathrm{d}y \\
&\overset{\circ}{=} \iint \|s_\theta(x,y)\|^2 p_{\mathsf{data}}(x,y)\mathrm{d}x\mathrm{d}y - 2\iint \nabla_x p_{\mathsf{data}}(x,y)^\top s_\theta(x,y)\mathrm{d}x\mathrm{d}y \\
&\overset{\circ}{=} \iint \|s_\theta(x,y)\|^2 p_{\mathsf{data}}(x,y)\mathrm{d}x\mathrm{d}y + 2\iint s_\theta(x,y)^\top \nabla_x [p_{\mathsf{data}}(y) p_{\mathsf{data}}(x|y)]\mathrm{d}x\mathrm{d}y \\
&\overset{\circ}{=} \iint \|s_\theta(x,y)\|^2 p_{\mathsf{data}}(x,y)\mathrm{d}x\mathrm{d}y + 2\iint s_\theta(x,y)^\top p_{\mathsf{data}}(y) \nabla_x [p_{\mathsf{data}}(x|y)]\mathrm{d}x\mathrm{d}y \\
&\overset{\circ}{=} \int \int \left\|s_\theta(x,y) - \nabla_x \log p_{\mathsf{data}}(x|y)\right\|^2 p_{\mathsf{data}}(x|y)\mathrm{d}x p_{\mathsf{data}}(y)\mathrm{d}y.
\end{aligned}
$$

If tractable the ESM objective would allows us to learn the
But we don't have access to the score $\nabla_x \log p_{\mathsf{data}}(x,y)$.

Conditional score matching (ctd.)

Suppose then that we instead look at

$$p^{\sigma}_{\text{data}}(x, y) = p_{\text{data}}(y) \int p_{\text{data}}(x - z|y) * \phi_{\sigma}(z)\mathrm{d}z,$$

where $*$ denotes convolution and $\phi_{\sigma}(z) = \mathcal{N}(z; 0, \sigma^2 \mathbb{1})$.

Then the same calculation as for unconditional denoising score matching

$$\int \left\| s_{\theta}(x, y) - \nabla_x \log p^{\sigma}_{\text{data}}(x|y) \right\|^2 p_{\text{data}}(x|y)\mathrm{d}x \overset{\circ}{=} \int \| s_{\theta}(x, y) - \nabla_x \log q_{\sigma}(z|x) \|^2 p_{\text{data}}(x|y) q_{\sigma}(z|x)\mathrm{d}z\mathrm{d}x$$

Therefore supposing we have samples $(X_i, Y_i) \sim p_{\text{data}}$ we can solve

$$\arg\min_{\theta} \sum_{i=1}^{n} \| s_{\theta}(X_i, Y_i) - \nabla_x \log q_{\sigma}(Z_i|X_i) \|^2, \qquad Z_i = X_i + \xi_i, \;\; \xi_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2 \mathbb{1})$$

Conditional Score Matching is already present in the original J. Song, Meng, and Ermon 2020 and Ho, Jain, and Abbeel 2020 without much detail.

Better explained in Batzolis et al. 2021.

Conditional DDMs

We can now learn the conditional score.

Given samples $(X_i, Y_i) \sim p_{\text{data}}$ we can train a conditional score network $s_\theta(x, y, t)$ by minimising for example the simplified objective

$$\arg\min_\theta := \sum_{i=1}^{N} \sum_{k=1}^{K} \left\| \xi^\theta \left(t_k, \sqrt{\bar{\alpha}_{t_k}} X_i + \sqrt{1 - \bar{\alpha}_{t_k}} \xi, Y_i\right) - \xi \right\|^2 \tag{11}$$

{ec

$$t_k \sim \text{Unif}(0 : T), \quad \xi \sim \mathcal{N}(0, I). \tag{12}$$

Discrete time: Given the trained model $\theta^\star$, we can sample from $\text{Law}(X|Y = y)$ by

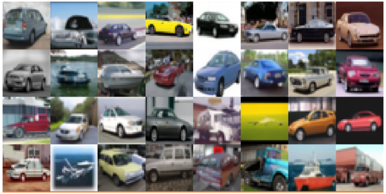
(1) Sample $X_T \sim \mathcal{N}(0, I)$

(2) For $t = T, T-1, \ldots, 1$ sample

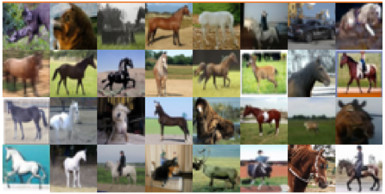
$$X_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(X_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \xi_{\theta^\star}(t, X_t, y) \right) + \sigma_t Z_t, \qquad Z_t \sim \mathcal{N}(0, I).$$

Conditional DDMs (ctd.)

For example Y could be a class label (e.g. horse, automobile), a low-resolution image, or a text embedding.



(a) Class-conditional image generation with DDMs conditioned on label car. From Ho, Jain, and Abbeel 2020.



(b) Class-conditional image generation with DDMs conditioned on label horse. From Ho, Jain, and Abbeel 2020.

Guidance

In generative models it is often useful to be able to do **lower temperature sampling**.

ie **reducing the randomness** in the generative process:

the aim is to **reduce diversity** but **improve sample quality**.

In DDMs this can be done by guidance.

There are two main types of guidance:

- Classifier guidance Dhariwal and A. Nichol 2021
- Classifier-free guidance Ho and Salimans 2021a

This is especially useful in conditional DDMs, where often classes overlap significantly.

Guidance

To understand the effect of guidance look at the following image from Ho and Salimans 2021b.

Figure: Effect of guidance on sample quality and diversity. From Ho and Salimans 2021b.

The leftmost image is a sample from the unconditional, ”true” model.

The remaining images are samples from mixtures of conditional models with increasing guidance.

Note: guidance samples from a different model.

Classifier guidance I

Classifier guidance was introduced in Dhariwal and A. Nichol 2021.
Ingredients:

- A conditional DDM with score $s_\theta(t, x, y)$ trained on samples $(X_i, Y_i) \sim p_{\text{data}}$.
- A classifier $p_t^\phi(y|x_t)$ trained to predict Y from **noisy samples** $X_t \sim q_{t|0}(\cdot|X)$.

Idea: use $\nabla_x \log p_t^\phi(y|x)$ to "guide" the diffusion towards samples that favour the class label y.
Algorithm: modify the reverse SDE as

$$\mathrm{d}X_t = -[X_t + 2\nabla \log s_t^\theta(X_t, y) + 2\gamma \nabla \log p_t^\phi(y|X_t)]\mathrm{d}t + \sqrt{2}\mathrm{d}B_t. \tag{13}$$

In discrete time this becomes, with the noise parameterisation,

$$X_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(X_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\xi_\theta(t, X_t, y)\right) + \sigma_t Z_t + \gamma\sigma_t^2 \nabla_x \log p_t^\phi(y|X_t). \tag{14}$$

Classifier guidance (ctd.)

Since:

- $s^\theta(t, x, y)$ is an approximation of $\nabla_x \log p_t(x|y)$;
- $p_t^\phi(y|x)$ is an approximation of $p_t(y|x_t)$;
- $p_t(y|x) = p_t(x|y)/p_t(x)$;

the SDE is really trying to approximate

$$\begin{aligned}
\mathrm{d}X_t &= -[X_t + 2\nabla_x \log p_t(X_t|y) + 2\gamma\nabla_x \log p_t^\phi(y|X_t)]\mathrm{d}t + \sqrt{2}\mathrm{d}B_t \\
&= -\left[X_t + 2\nabla_x \log\left(p_t(X_t|y)p_t(y|X_t)^\gamma\right)\right]\mathrm{d}t + \sqrt{2}\mathrm{d}B_t
\end{aligned}$$

BUT: it is wrong to interpret this as sampling from $p_t(x|y)p_t(y|x)^\gamma$, Chidambaram et al. 2024; Bradley and Nakkiran 2024.

Strength γ: controls the amount of guidance. Increasing γ reduces diversity but improves sample quality. Can be interpreted as corresponding to a classifier $p(y|x)^\gamma$.

Classifier guidance (ctd.)

Figure: Effect of guidance strength γ on sample quality and diversity. Class label is "Pembroke Welsh Corgi". Left $\gamma = 1.0$, right $\gamma = 10.0$. From Dhariwal and A. Nichol 2021.

Classifier-free guidance

Classifier-free guidance was introduced in Ho and Salimans (2021b).

Ingredients:

– A conditional DDM $p^\theta(x|y)$ trained on samples $(X_i, Y_i) \sim p_{\text{data}}$.

Idea: learn an unconditional model $p^\theta(x)$ by **randomly dropping** the condition Y during training.

That is we learn a single model $s^\theta(x, y, t)$, where y is either the class label or a special token $\varnothing$ indicating no class.

This way we learn **both** the conditional and unconditional score.

Algorithm: modify the reverse SDE as

$$\mathrm{d}X_t = -[X_t + 2(1+\gamma)\nabla \log s_t^\theta(X_t, y) - 2\gamma \nabla \log s_t^\theta(X_t)]\mathrm{d}t + \sqrt{2}\mathrm{d}B_t. \tag{15}$$

Flow matching

Flow matching

Flow matching was introduced in Lipman et al. 2023.

The basic objective remains the same:

given samples $X_1, \ldots, X_n \sim p_{\mathsf{data}}$ learn to sample from p_{data}.

The major difference form diffusion models is that **there is no noise** in the latent variable.

Instead it learns a deterministic map $\phi : \mathbb{R}^d \mapsto \mathbb{R}^d$ such that if $Z \sim \mathcal{N}(0, I)$ then

Equivalently it learns a map ϕ such that

$$\phi^{\#}\mathcal{N}(0, I) = p_{\mathsf{data}}.$$

Although this is essentially a normalising flow, flow matching is much more flexible as we will see.

A quick intro to Normalising flows

Normalising flows attempt learn a map ϕ such that

$$\phi^{\#}\mathcal{N}(0, I) = p_{\mathsf{data}}.$$

The map will belong to a parametric family ϕ_θ, that could be a class of neural networks or compositions thereof.

Normalising flows are typically trained by maximum likelihood:

$$\arg\max_\theta \sum_{i=1}^{n} \log p^\theta(X_i), \qquad X_i \sim p_{\mathsf{data}}$$

where p^θ is the density of $\phi_\theta^{\#}\mathcal{N}(0, I)$.

Here is the catch: we need to evaluate p^θ, but it is typically **intractable**.

Our only hope is to use the change of variables formula:

$$p^\theta(x) = \rho(\phi_\theta^{-1}(x)) \left|\det \nabla \phi_\theta^{-1}(x)\right|, \tag{16}$$

where ρ is the density of $\mathcal{N}(0, I)$.
But this only holds if ϕ_θ is invertible and differentiable with tractable Jacobian determinant.

Continuous Normalising flows

We are more interested in continuous normalising flows R. T. Chen et al. 2018.

Instead of sending $\mathcal{N}(0,1)$ directly to p_{data} we construct a:

path of distributions $(\rho_t : t \in [0,1])$ such that $\rho_0 = \mathcal{N}(0,I)$ and $\rho_1 = p_{\text{data}}$.

This is done through time-dependent vector field $u : [0,1] \times \mathbb{R}^d \mapsto \mathbb{R}^d$.

The vector field defines a flow through the ODE

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = u(t, X_t), \qquad X_0 \sim \rho_0. \tag{17}$$

Let $\phi_t(x) = \phi_t^u(x)$ be the solution at time t of the ODE with initial condition $X_0 = x$.

R. T. Chen et al. 2018 suggested modelling ϕ with a neural network ϕ_θ and training it by maximum likelihood.

Maximum likelihood can be used to train CNFs based on the instantaneous change of variables formula:

$$[\phi_t]^{\#}\rho_0(x) = \rho_t(x) = \rho_0(\phi_t^{-1}(x)) \det\left[\frac{\partial \phi_t^{-1}}{\partial x}(x)\right]. \tag{18}$$

Flow matching

Flow matching Lipman et al. 2023 takes a different approach.
It directly defines a path of distributions $(\rho_t : t \in [0,1])$, with $\rho_0 = \mathcal{N}(0, I)$ (say) and $\rho_1 = p_{\text{data}}$, as follows:

Let $X_0 \sim \rho_0$ and $X_1 \sim p_{\text{data}}$ and define

$$\rho_t = \text{Law}(X_t), \qquad \text{where } X_t := (1-t)X_0 + tX_1, \qquad t \in [0,1]. \tag{19}$$

Problem: this is **non-causal**, we need to know the end point to simulate.

Question: can we learn a vector field $u_\theta(t,x)$ such that $\phi_t^{u_\theta} \# \rho_0 = \rho_t$?

The answer is yes, and the key is the continuity equation.

Flow matching (ctd.)

Suppose that there exists a vector field $u(t,x)$ such that $\phi_t^u \# \rho_0 = \rho_t$.

Then ρ_t satisfies the **continuity equation**

$$\partial_t \rho_t(x) + \nabla \cdot (\rho_t(x) u(t,x)) = 0. \tag{20}$$

Let us approximate u with a neural network $u_\theta(t,x)$ by solving

$$\arg\min_\theta \int_0^1 \mathbb{E}_{X_t \sim \rho_t} \left[\| u(t,X_t) - u_\theta(t,X_t) \|^2 \right] \mathrm{d}t. \tag{FM}$$

Flow matching (ctd.)

Then

$$\int_0^1 \mathbb{E}_{X_t \sim \rho_t}\left[\|u(t,X_t) - u_\theta(t,X_t)\|^2\right] \mathrm{d}t$$

$$\overset{\circ}{=} \int_0^1 \mathbb{E}_{X_t \sim \rho_t}\left[\|u(t,X_t) - u_\theta(t,X_t)\|^2\right] \mathrm{d}t$$

since $u(t, X_t) = \mathrm{d}X_t/\mathrm{d}t$

$$\overset{\circ}{=} \int_0^1 \mathbb{E}_{X_t \sim \rho_t}\left\|\frac{\mathrm{d}X_t}{\mathrm{d}t} - u_\theta(t,X_t)\right\|^2 \mathrm{d}t$$

since $X_t = (1-t)X_0 + tX_1 = X_0 + t(X_1 - X_0)$

$$\overset{\circ}{=} \int_0^1 \mathbb{E}_{X_t \sim \rho_t}\|X_1 - X_0 - u_\theta(t,X_t)\|^2 \mathrm{d}t$$

Flow matching (ctd.)

In particular, an exact solution is given by

$$u(t,x) = \mathbb{E}[X_1 - X_0 | X_t = x].$$

Remark: this is the conditional expectation of a random variable,

Theorem (Lipman et al. 2023)

Let $X_0 \sim \rho_0$ and $X_1 \sim \rho_1$ be independent random variables. Define $X_t = (1-t)X_0 + tX_1$ and $\rho_t = \mathrm{Law}(X_t)$ for $t \in [0,1]$. Suppose that $u(t,x) = \mathbb{E}[X_1 - X_0 | X_t = x]$ is well defined and continuous. Then $\phi_t^u \# \rho_0 = \rho_t$.

Flow matching (ctd.)

Besides being elegant, the fact that $\mathbb{E}[X_1 - X_0 | X_t = x]$ does what we want has great practical advantages.

In the scenario of interest, we have access to i.i.d. samples $X_1^i \sim p_{\mathsf{data}}$, whereas it is by choice easy to sample $X_0^i \sim \rho_0$.

We simply pair them up randomly and solve

$$\arg\min_\theta \sum_{i=1}^n \int_0^1 \mathbb{E} \left\| X_1^i - X_0^i - u_\theta(t, X_t^i) \right\|^2 \mathrm{d}t, \tag{21}$$

where $X_t^i = (1-t)X_0^i + tX_1^i$.

Flow matching (ctd.) I

Algorithm: given samples $X_i \sim p_{\text{data}}$ and $Z_i \sim \mathcal{N}(0, I)$ we can train a vector field $u_\theta(t, x)$ by solving

$$\arg\min_\theta \sum_{i=1}^{n} \int_0^1 \mathbb{E}_{\xi \sim \mathcal{N}(0,I)} \left\| X_i - Z_i - u_\theta(t, (1-t)Z_i + tX_i + \sqrt{t(1-t)}\xi) \right\|^2 \mathrm{d}t. \tag{22}$$

where the noise ξ is added to ensure that the distribution of $(1-t)Z_i + tX_i + \sqrt{t(1-t)}\xi$ has full support.

Once we have trained u_θ we can sample from p_{data} by solving the ODE

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = u_\theta(t, X_t), \qquad X_0 \sim \mathcal{N}(0, I).$$

Choosing the path

The choice of path we gave in the earlier slide is only one of many possible.

We can construct such paths as mixtures of simpler paths, that may even be defined per sample.

One idea presented in Lipman et al. 2023 and Albergo and Vanden-Eijnden 2023 is to condition the path on a **latent variable** Z.

This could actually be the data point itself, ie $Z = X_1 \sim p_{\text{data}}$.

One then builds a **conditional** path $\rho_{t|z}$ so that

$$\rho_t = \int \rho_{t|z} p_{\text{data}}(z) \mathrm{d}z.$$

Benefit: $\rho_{t|z}$ can be much simpler to design, tractable and easy to sample from.

In order for ρ_t to interpolate between ρ_0 and $\rho_1 := p_{\text{data}}$ we need $\rho_{t|z}$ to satisfy **boundary conditions**, e.g.

$$\rho_0(x|z) = \rho_0, \qquad \rho_1(x|z) = \mathcal{N}(x; z, \sigma_{\text{min}}^2 I) \approx \delta_z.$$

ρ_0 can be any simple distribution, e.g. $\mathcal{N}(0, I)$.

Continuity/transport equation

We will need the following well-known result from the theory of continuity/transport equations.

Theorem

Suppose that

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = u(t, X_t), \qquad X_0 \sim \rho_0 \tag{23}$$

Then, if $\rho_t = \mathsf{Law}(X_t)$*, it satisfies the continuity equation*

$$\frac{\mathrm{d}\rho_t}{\mathrm{d}t} + \nabla \cdot (\rho_t u(t, \cdot)) = 0. \tag{24}$$

Continuity/transport equation (ctd.)

Proof.

The proof is fairly straightforward and quite useful to understand. Let $\varphi \in C_c^\infty(\mathbb{R}^d)$ be a smooth compactly supported test function.
Then

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int \varphi(x)\rho_t(x)\mathrm{d}x &= \frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[\varphi(X_t)] = \frac{\mathrm{d}}{\mathrm{d}t}\int \rho_t(x)\varphi(x)\mathrm{d}x \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\int \varphi(X_t(x))\rho_0(x)\mathrm{d}x = \int \nabla\varphi(X_t(x))\frac{\mathrm{d}X_t(x)}{\mathrm{d}t}\rho_0(x)\mathrm{d}x \\ &= \int \nabla\varphi(X_t(x))^\top u(t, X_t)\rho_0(x)\mathrm{d}x = \int \nabla\varphi(x)^\top u(t,x)\rho_t(x)\mathrm{d}x\end{aligned}$$

and using integration by parts

$$= -\int \varphi(x)^\top \nabla \cdot \Big[u(t,x)\rho_t(x)\Big]\mathrm{d}x.$$

Flow matching with conditional paths continued

Suppose that we have constructed a conditional path $\rho_{t|z}$ satisfying the boundary conditions.

Suppose also that the conditional probability path satisfies the continuity equation

$$\frac{\mathrm{d}\rho_{t|z}}{\mathrm{d}t} + \nabla \cdot (\rho_{t|z} u(t, \cdot|z)) = 0. \tag{25}$$

Lipman et al. 2023 noticed that we can use the **conditional vector field** $u(t, x|z)$ to express the marginal vector field $u(t, x)$ that drives the continuity equation of the marginal path ρ_t:

$$u(t, x) = \mathbb{E}_{Z\sim p_{\mathsf{data}}}[u(t, x|Z)|X_t = x] = \int u(t, x|z) \frac{\rho_{t|z}(x) p_{\mathsf{data}}(z)}{\rho_t(x)} \mathrm{d}z. \tag{26}$$

{ec

Flow matching with conditional paths continued

Proof.

It suffices to check that $u(t,x)$ satisfies the continuity equation for ρ_t:

$$
\begin{aligned}
&\frac{\mathrm{d}\rho_t}{\mathrm{d}t} + \nabla_x \cdot (\rho_t(x)u(t,x)) \\
&= \frac{\mathrm{d}}{\mathrm{d}t}\int \rho_{t|z}(x)p_{\mathsf{data}}(z)\mathrm{d}z + \nabla_x \cdot \left(\int \rho_{t|z}(x)p_{\mathsf{data}}(z)u(t,x|z)\mathrm{d}z\right) \\
&= \int \left[\frac{\mathrm{d}\rho_{t|z}}{\mathrm{d}t} + \nabla_x \cdot (\rho_{t|z}(x)u(t,x|z))\right] p_{\mathsf{data}}(z)\mathrm{d}z = 0.
\end{aligned}
$$

$\square$

Conditional flow matching objective

The marginal vector field $u(t,x)$ involves an intractable expectation.

Therefore the **flow matching objective**

$$\arg\min_{\theta} \int_0^1 \mathbb{E}_{X_t \sim \rho_t} \left[\| u(t, X_t) - u_\theta(t, X_t) \|^2 \right] \mathrm{d}t \tag{FM}$$

is also intractable.

Lipman et al. 2023 showed we can bypass this expectation and instead use the **tractable conditional flow matching objective** (CFM):

$$\arg\min_{\theta} \int_0^1 \mathbb{E}_{Z \sim p_{\text{data}}} \mathbb{E}_{X_t \sim \rho_{t|Z}} \left[\| u(t, X_t|Z) - u_\theta(t, X_t|Z) \|^2 \right] \mathrm{d}t. \tag{CFM}$$

Since we typically have chosen $u(t,x|z)$ to be tractable, and we have access to samples from $\rho_{t|Z}$, we can estimate CFM objective.

Conditional flow matching objective

FACT: minimising CFM minimises the FM . Using (26) we have

$$
\begin{aligned}
&\int_0^1 \mathbb{E}_{X_t \sim \rho_t}\left[\|u(t, X_t) - u_\theta(t, X_t)\|^2\right] \mathrm{d}t \\
&\overset{\circ}{=} \int_0^1 \int \left[\|u_\theta(t, x)\|^2 - 2u_\theta(t, x)^\top u(t, x)\right] \rho_t(x) \mathrm{d}x \mathrm{d}t \\
&\overset{\circ}{=} \int_0^1 \int \left[\|u_\theta(t, x)\|^2 - 2 \int u(t, x|z) \frac{\rho_{t|z}(x) p_{\mathsf{data}}(z)}{\rho_t(x)} \mathrm{d}z\right] \rho_t(x) \mathrm{d}x \mathrm{d}t \\
&\overset{\circ}{=} \int_0^1 \int \int \left[\|u_\theta(t, x)\|^2 - 2u_\theta(t, x)^\top u(t, x|z)\right] \cancel{\rho_t(x)} \frac{\rho_{t|z}(x) p_{\mathsf{data}}(z)}{\cancel{\rho_t(x)}} \mathrm{d}z \mathrm{d}x \mathrm{d}t \\
&\overset{\circ}{=} \int_0^1 \int \int \left[u_\theta(t, x) - u(t, x|z)\right]^2 \rho_{t|z}(x) p_{\mathsf{data}}(z) \mathrm{d}z \mathrm{d}x \mathrm{d}t \\
&= \int_0^1 \mathbb{E}_{\substack{Z \sim p_{\mathsf{data}} \\ X_t \sim \rho_{t|Z}}} \left[\|u(t, X_t|Z) - u_\theta(t, X_t|Z)\|^2\right] \mathrm{d}t.
\end{aligned}
$$

Theory for DDMs

Theory for DDMs

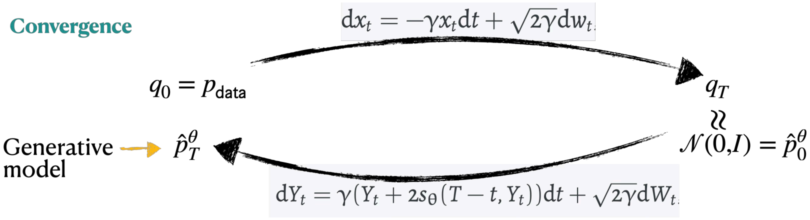


Figure: Schematic of generative model.

Obvious question: how close is p_T^θ to p_{data}?

Convergence of DDMs

IF we had access to the true score $\nabla_x \log p_t(x)$ and could initialise from q_T and solve the reverse SDE exactly

$$\mathrm{d}X_t = -[X_t + 2\nabla \log p_t(X_t)]\mathrm{d}t + \sqrt{2}\mathrm{d}B_t, \qquad X_T \sim q_T,$$

then we would have exact sampling from p_{data} at time $t = 0$.
In practice there are three sources of error:

(a) **Score estimation error:** we only have an approximation $s_\theta(t, x)$ of the score.

(b) **Discretisation error:** we discretise the SDE.

(c) **Initialisation error:** we initialise from $\mathcal{N}(0, I)$ instead of q_T.

Attempt 1: treat error 1 as a black box, assuming

Bound error assuming

$$\int_0^T \mathbb{E}_{X_t \sim q_t}\left[\|s_\theta(t, X_t) - \nabla_x \log q_t(X_t)\|^2\right] \mathrm{d}t \leqslant \varepsilon_{\text{score}}^2. \qquad (27)$$

{eq:sc

Convergence of DDMs (ctd.)

Early results:

- **Restrictive assumptions:** e.g. log-Sobolev, Lipschitz score Lee, Lu, and Tan 2023; K. Y. Yang and Wibisono 2022
- **non quantitative:** e.g. Pidstrigach 2022
- **exponential dependence on parameters:** e.g. dimension, Bortoli 2022; Block, Mroueh, and Rakhlin 2022

More recent results:

polynomial bounds: H. Chen, Lee, and Lu 2023; Lee, Lu, and Tan 2023

Linear in dimension: for **Lipschitz scores**, but Lipschitz constant hides dimensionality factors.

Quadratic in d for general distributions.

Convergence of DDMs (ctd.)

First **linear in d** result for general case appeared in Benton et al. 2024.

Optimal in general.

BUT, what happens if data lives in a lower-dimensional manifold?

- Ambient dimension very high in typical image datasets
- Still too high to explain success of diffusion models
- Possible explanation: concentrates on low-dimensional subset with structure
- This is known as the **manifold hypothesis**

Manifold hypothesis

Many modern datasets are conjectured to satisfy a form of the **manifold hypothesis**

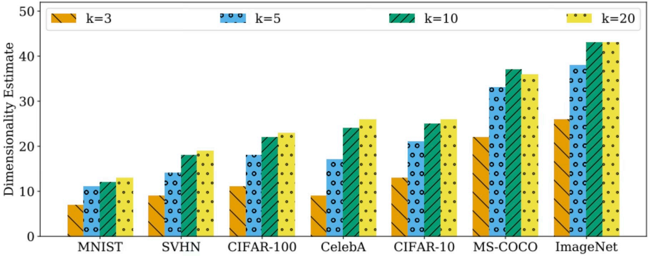


Figure: Estimates of intrinsic dimension of popular image datasets. From Pope et al, arxiv:2104.08894.

Convergence under manifold hypothesis

Theorem (Potaptchik, Azangulov, and Deligiannidis 2025)

Suppose that p_{data} is supported on a $d^\star$-dimensional smooth compact manifold $\mathcal{M} \subset \mathbb{R}^d$. Then under smoothness assumptions on p_{data}[a] and the manifold, we have

$$\mathsf{K}(\hat{Y}_{T-\delta} \| X_\delta) \lesssim \varepsilon^2 + \varepsilon^2_{\text{score}} + \frac{d^*(\log \delta^{-1} + \log \varepsilon^{-1} + \log d)(\log \delta^{-1} + C) \log \delta^{-1}}{K}. \tag{28}$$

[a] density wrt Hausdorff measure on $\mathcal{M}$

And what about the score estimator?

Problem: how to ensure that the score estimator satisfies (27) with small $\varepsilon_{\text{score}}$?

This is a tough problem and not solved fully yet. In particular theory cannot account for the role of the **architecture** (e.g. UNets, convolutional networks) or **training algorithms** (e.g. SGD).

Some results do exist however, where the neural network is selected from a carefully constructed class through **ERM**

- Oko, Akiyama, and Suzuki 2023— p_{data} full support; rates minimax
- Tang and Y. Yang 2024— p_{data} lives in a $d^\star$ manifold, almost minimax rates in $d^\star$ but large polynomial constants in D;
- Azangulov, Deligiannidis, and Rousseau 2025— p_{data} lives in a $d^\star$ manifold, almost minimax rates in $d^\star$, no explicit dependence on D for score estimation, $\sqrt{D}$ for sampling.

Hot topics for research

Discrete space DDMS: use diffusion instead of autoregressive in LLMs Li et al. 2025; Sahoo et al. 2024 and many more.

Protein structure prediction: Watson et al. 2023 used diffusion models to generate protein structures. David Baker awarded 2024 Nobel prize.

Accelerating/Distilling Diffusion models: good results with less expensive inference steps: e.g. Bortoli et al. 2025; Yin et al. 2024; Frans et al. 2024

Thank you for your attention!

References I

[1] Michael Samuel Albergo and Eric Vanden-Eijnden. "Building Normalizing Flows with Stochastic Interpolants". In: *The Eleventh International Conference on Learning Representations*. 2023. URL: https://openreview.net/forum?id=li7qeBbCR1t.

[2] Brian DO Anderson. "Reverse-time diffusion equation models". In: *Stochastic Processes and their Applications* 12.3 (1982), pp. 313–326.

[3] Iskander Azangulov, George Deligiannidis, and Judith Rousseau. *Convergence of Diffusion Models Under the Manifold Hypothesis in High-Dimensions*. 2025. arXiv: 2409.18804 [stat.ML]. URL: https://arxiv.org/abs/2409.18804.

[4] Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. "Conditional image generation with score-based diffusion models". In: *arXiv preprint arXiv:2111.13606* (2021).

References II

[5] Joe Benton, Valentin De Bortoli, Arnaud Doucet, and George Deligiannidis. "Nearly d-Linear Convergence Bounds for Diffusion Models via Stochastic Localization". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=r5njV3BsuD.

[6] Adam Block, Youssef Mroueh, and Alexander Rakhlin. *Generative Modeling with Denoising Auto-Encoders and Langevin Sampling*. 2022. arXiv: 2002.00107 [stat.ML]. URL: https://arxiv.org/abs/2002.00107.

[7] Valentin De Bortoli. "Convergence of denoising diffusion models under the manifold hypothesis". In: *Transactions on Machine Learning Research* (2022). Expert Certification. ISSN: 2835-8856.

[8] Valentin De Bortoli, Alexandre Galashov, J Swaroop Guntupalli, Guangyao Zhou, Kevin Patrick Murphy, Arthur Gretton, and Arnaud Doucet. "Distributional Diffusion Models with Scoring Rules". In: *Forty-second International Conference on Machine Learning*. 2025. URL: https://openreview.net/forum?id=N82967FcVK.

References III

[9] Arwen Bradley and Preetum Nakkiran. "Classifier-free guidance is a predictor-corrector". In: *arXiv preprint arXiv:2408.09000* (2024).

[10] Hongrui Chen, Holden Lee, and Jianfeng Lu. "Improved analysis of score-based generative modeling: User-friendly bounds under minimal smoothness assumptions". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 4735–4763.

[11] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. "Neural ordinary differential equations". In: *Advances in neural information processing systems* 31 (2018).

[12] Muthu Chidambaram, Khashayar Gatmiry, Sitan Chen, Holden Lee, and Jianfeng Lu. "What does guidance do? a fine-grained analysis in a simple setting". In: *arXiv preprint arXiv:2409.13074* (2024).

[13] Prafulla Dhariwal and Alexander Nichol. "Diffusion models beat gans on image synthesis". In: *Advances in Neural Information Processing Systems* 34 (2021).

References IV

[14] Bradley Efron. “Tweedie’s formula and selection bias”. In: *Journal of the American Statistical Association* 106.496 (2011), pp. 1602–1614.

[15] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. “One step diffusion via shortcut models”. In: *arXiv preprint arXiv:2410.12557* (2024).

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[17] Jonathan Ho and Tim Salimans. “Classifier-Free Diffusion Guidance”. In: *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. 2021.

[18] Jonathan Ho and Tim Salimans. “Classifier-Free Diffusion Guidance”. In: *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. 2021. URL: https://openreview.net/forum?id=qw8AKxfYbI.

[19] Aapo Hyvärinen. “Estimation of non-normalized statistical models by score matching.”. In: *Journal of Machine Learning Research* 6.4 (2005).

References V

[20] Holden Lee, Jianfeng Lu, and Yixin Tan. "Convergence of score-based generative modeling for general data distributions". In: *International Conference on Algorithmic Learning Theory*. PMLR. 2023, pp. 946–985.

[21] Tianyi Li, Mingda Chen, Bowei Guo, and Zhiqiang Shen. "A survey on diffusion language models". In: *arXiv preprint arXiv:2508.10875* (2025).

[22] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. "Flow Matching for Generative Modeling". In: *The Eleventh International Conference on Learning Representations*. 2023.

[23] Alexander Quinn Nichol and Prafulla Dhariwal. "Improved denoising diffusion probabilistic models". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8162–8171.

[24] Kazusato Oko, Shunta Akiyama, and Taiji Suzuki. "Diffusion models are minimax optimal distribution estimators". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 26517–26582.

References VI

[25] Jakiw Pidstrigach. “Score-based generative models detect manifolds”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 35852–35865.

[26] Peter Potaptchik, Iskander Azangulov, and George Deligiannidis. “Linear Convergence of Diffusion Models Under the Manifold Hypothesis”. In: *Proceedings of Thirty Eighth Conference on Learning Theory*. Ed. by Nika Haghtalab and Ankur Moitra. Vol. 291. Proceedings of Machine Learning Research. 30 Jun–04 Jul 2025, pp. 4668–4685.

[27] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. *Simple and Effective Masked Diffusion Language Models*. 2024. arXiv: 2406.07524 [cs.CL]. URL: https://arxiv.org/abs/2406.07524.

[28] Jiaming Song, Chenlin Meng, and Stefano Ermon. “Denoising diffusion implicit models”. In: *arXiv preprint arXiv:2010.02502* (2020).

[29] Yang Song and Stefano Ermon. “Generative modeling by estimating gradients of the data distribution”. In: *NeurIPS* (2019).

References VII

[30] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. “Score-based generative modeling through stochastic differential equations”. In: *ICLR* (2021).

[31] Rong Tang and Yun Yang. “Adaptivity of diffusion models to manifold structures”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2024, pp. 1648–1656.

[32] Pascal Vincent. “A connection between score matching and denoising autoencoders”. In: *Neural Computation* 23.7 (2011), pp. 1661–1674.

[33] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. “De novo design of protein structure and function with RFdiffusion”. In: *Nature* 620.7976 (2023), pp. 1089–1100.

References VIII

[34] Kaylee Yingxi Yang and Andre Wibisono. "Convergence of the inexact Langevin algorithm and score-based generative models in KL divergence". In: *arXiv preprint arXiv:2211.01512* (2022).

[35] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. "One-step diffusion with distribution matching distillation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 6613–6623.