Diffusion Models Chapter 2: Training via Score Matching

Generative AI and Foundation Models

Spring 2024

Department of Mathematical Sciences

Ernest K. Ryu

Seoul National University

Practical reverse-time SDE

Simulating the reverse-time SDE

$$d\overline{X}_t = (f - g^2 \nabla \log p_t)dt + g d\bar{W}_t, \qquad \overline{X}_T \sim p_T$$

requires (i) sample from p_T and (ii) evaluations of the *score function*[#] $\nabla_x \log p_t$.

Solution:

(i) Design forward-time SDE, i.e., choose f, g, T, so that $p_T \approx \mathcal{N}(0, \sigma_T^2 I)$ and σ_T^2 is known.

(ii) Learn $\nabla_x \log p_t(x) \approx s_\theta(x, t)$ via a neural network $s_\theta(x, t)$.
We call $s_\theta(x, t)$ the *score network*.

[#]Some call this the Stein score function, while some people argue that the name "score function" is confusing and should not be used as the Fisher score function is a similar but different object. In any case, DPM papers universally refer to this as the "score function".

Score matching

To learn the score function, consider

$$\mathcal{L}(\theta) = \int_0^T \lambda(t)\mathbb{E}_{X_t}[\|s_\theta(X_t, t) - \nabla_{X_t} \log p_t(X_t)\|^2]dt$$

where $\lambda(t) > 0$ is a weighing factor. However, we cannot use this as is, since p_t is inaccessible.

Alternatively, use the equivalent losses

1. $$\mathcal{L}(\theta) = \int_0^T \lambda(t)\mathbb{E}_{X_0}\left[\mathbb{E}_{X_t|X_0}[\|s_\theta(X_t, t) - \nabla_{X_t} \log p_{t|0}(X_t \mid X_0)\|^2 \mid X_0]\right] dt + C$$

2. $$\mathcal{L}(\theta) = \int_0^T \lambda(t)\mathbb{E}_{X_t}\left[\|s_\theta(X_t, t)\|^2 + 2\mathbb{E}_\nu\left[\frac{d}{dh}\nu^\intercal s_\theta(X_t + h\nu, t)\Big|_{h=0}\right]\right] dt + C$$

where, C are constants independent of θ.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, Score-based generative modeling through stochastic differential equations, *ICLR*, 2021.

proof

(1) $\mathcal{L}(\theta) = \int_0^T \lambda(t)\mathbb{E}_{X_0}\left[\mathbb{E}_{X_t|X_0}[\|s_\theta(X_t,t) - \nabla_{X_t}\log p_{t|0}(X_t \mid X_0)\|^2 \mid X_0]\right] dt + C$

The replacement of $\nabla_{X_t}\log p_t(X_t)$ with $\nabla_{X_t}\log p_{t|0}(X_t|X_0)$ requires justification.

$$
\begin{aligned}
\nabla_{X_t}\log p_t(X_t) &= \frac{\nabla_{X_t} p_t(X_t)}{p_t(X_t)} \\
&= \frac{1}{p_t(X_t)}\nabla_{X_t}\int_{\mathbb{R}^d} p_{t|0}(X_t|X_0)p_0(X_0)\, dX_0 \\
&= \int_{\mathbb{R}^d}\left(\nabla_{X_t} p_{t|0}(X_t|X_0)\right)\frac{p_0(X_0)}{p_t(X_t)}\, dX_0 \\
&= \int_{\mathbb{R}^d}\left(\nabla_{X_t}\log p_{t|0}(X_t|X_0)\right)\frac{p_0(X_0)p_{t|0}(X_t|X_0)}{p_t(X_t)}\, dX_0 \\
&= \int_{\mathbb{R}^d}\left(\nabla_{X_t}\log p_{t|0}(X_t|X_0)\right) p_{0|t}(X_0|X_t)\, dX_0 \\
&= \mathbb{E}_{X_0|X_t}\left[\nabla_{X_t}\log p_{t|0}(X_t|X_0) \mid X_t\right]
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L}(\theta) &= \int_0^T \lambda(t)\mathbb{E}_{X_t}[\|s_\theta(X_t,t) - \nabla_{X_t}\log p_t(X_t)\|^2]\, dt \\
&= \int_0^T \lambda(t)\mathbb{E}_{X_t}[\|s_\theta(X_t,t)\|^2 - 2\langle s_\theta(X_t,t), \nabla_{X_t}\log p_t(X_t)\rangle]\, dt + C \\
&= \int_0^T \lambda(t)\mathbb{E}_{X_t}[\|s_\theta(X_t,t)\|^2 - 2\left\langle s_\theta(X_t,t), \mathbb{E}_{X_0|X_t}[\nabla_{X_t}\log p_{t|0}(X_t \mid X_0) \mid X_t]\right\rangle]\, dt + C \\
&= \int_0^T \lambda(t)\mathbb{E}_{X_t}\left[\mathbb{E}_{X_0|X_t}[\|s_\theta(X_t,t)\|^2 - 2\left\langle s_\theta(X_t,t), \nabla_{X_t}\log p_{t|0}(X_t \mid X_0)\right\rangle \mid X_t]\right]\, dt + C \\
&= \int_0^T \lambda(t)\mathbb{E}_{X_t,X_0}[\|s_\theta(X_t,t) - \nabla_{X_t}\log p_{t|0}(X_t \mid X_0)\|^2]\, dt + C
\end{aligned}
$$

Called *denoising score matching (DSM)*.

P. Vincent, A connection between score matching and denoising autoencoders, *Neural Computation*, 2011.

proof

(1) $\mathcal{L}(\theta) = \int_0^T \lambda(t)\mathbb{E}_{X_0}\left[\mathbb{E}_{X_t|X_0}[\|s_\theta(X_t,t) - \nabla_{X_t}\log p_{t|0}(X_t \mid X_0)\|^2 \mid X_0]\right] dt$

Conditional score function is implementable if f and g are nice.

Ornstein–Uhlenbeck process is one such example.

$$dX_t = -\beta X_t dt + \sigma dW_t$$

$$p_{t|0}(X_t \mid X_0) \sim \mathcal{N}\left(e^{-\beta t}X_0, \sigma_t^2 I\right), \qquad \sigma_t^2 = \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})$$

$$\nabla_{X_t}\log p_{t|0}(X_t \mid X_0) = \frac{1}{\sigma_t^2}(X_t - e^{-\beta t}X_0)$$

$$= \frac{2\beta}{\sigma^2(1 - e^{-2\beta t})}(X_t - e^{-\beta t}X_0)$$

Hutchinson's trace estimator

Let $\nu \in \mathbb{R}^n$ be a random vector such that

$$\mathbb{E}_\nu[\nu_i \nu_j] = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

i.e., $\mathbb{E}_\nu[\nu\nu^\intercal] = I \in \mathbb{R}^{n \times n}$.

One example is $\nu_1, \ldots, \nu_n \sim \mathcal{N}(0,1)$ IID Gaussian.

Another example is $\nu_1, \ldots, \nu_n$ drawn as IID Rademacher (± 1 realization with probability $1/2$) random variables.

M. F. Hutchinson, A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines, *Communications in Statistics - Simulation and Computation*, 1990.

Hutchinson's trace estimator

Let $A \in \mathbb{R}^{n \times n}$. Then

$$\begin{aligned}
\mathbb{E}_\nu[\nu^\intercal A \nu] &= \mathbb{E}_\nu[\mathrm{Tr}(\nu^\intercal A \nu)] \\
&= \mathbb{E}_\nu[\mathrm{Tr}(A \nu \nu^\intercal)] \\
&= \mathrm{Tr}\left(\mathbb{E}_\nu[A \nu \nu^\intercal]\right) \\
&= \mathrm{Tr}\left(A \mathbb{E}_\nu[\nu \nu^\intercal]\right) \\
&= \mathrm{Tr}\left(AI\right) \\
&= \mathrm{Tr}(A)
\end{aligned}$$

So

$$\mathrm{Tr}(A) = \mathbb{E}_\nu[\nu^\intercal A \nu]$$

and $\nu^\intercal A \nu$ serves as an unbiased estimator of $\mathrm{Tr}(A)$.

M. F. Hutchinson, A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines, *Communications in Statistics - Simulation and Computation*, 1990.

proof

(2) $\mathcal{L}(\theta) = \int_0^T \lambda(t)\mathbb{E}_{X_t}\left[\|s_\theta(X_t,t)\|^2 + 2\mathbb{E}_\nu\left[\frac{d}{dh}\nu^\intercal s_\theta(X_t+h\nu,t)\Big|_{h=0}\right]\right]dt + C$

$$
\begin{aligned}
-\mathbb{E}_{X_t}\left[\langle s_\theta(X_t,t), \nabla_{X_t}\log p_t(X_t)\rangle\right] &= -\int \left\langle s_\theta(x,t), \frac{\nabla_x p_t(x)}{p_t(x)}\right\rangle p_t(x)dx \\
&= -\int \langle s_\theta(x,t), \nabla_x p_t(x)\rangle\, dx \\
&= \int (\nabla\cdot s_\theta(x,t))p_t(x)dx \\
&= \mathbb{E}_{X_t\sim p_t}[\nabla_{X_t}\cdot s_\theta(X_t,t))] \\
&= \mathbb{E}_{X_t}[\mathrm{Tr}\left(D_{X_t}s_\theta(X_t,t)\right)] \\
&= \mathbb{E}_{X_t}\mathbb{E}_\nu\left[\nu^\intercal D_{X_t}s_\theta(X_t,t)\nu\right] \\
&= \mathbb{E}_{X_t}\mathbb{E}_\nu\left[\frac{d}{dh}\nu^\intercal s_\theta(X_t+h\nu,t)\Big|_{h=0}\right]
\end{aligned}
$$

where we use integration by parts and the Hutchinson estimator.
Called *sliced score matching (SSM)*.

VE and VP forward SDEs

Two types Ornstein–Uhlenbeck processes are primarily considered for the forward SDE.

First, variance-exploding (VE)

$$dX_t = \sigma dW_t \qquad \gamma_t = 1$$
$$X_t \,|\, X_0 \sim \mathcal{N}\left(\gamma_t X_0, \sigma_t^2 I\right) \qquad \sigma_t^2 = t\sigma^2$$

Second, variance-preserving (VP)

$$dX_t = -\beta X_t dt + \sigma dW_t \qquad \gamma_t = e^{-\beta t}$$
$$X_t \,|\, X_0 \sim \mathcal{N}\left(\gamma_t X_0, \sigma_t^2 I\right) \qquad \sigma_t^2 = \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})$$

In both cases,

$$X_t \overset{\mathcal{D}}{=} \gamma_t X_0 + \sigma_t \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I)$$

General VE SDE

Let σ_t be a non-decreasing function of t.

General variance exploding (VE) SDE:

$$dX_t = \sqrt{\frac{d(\sigma_t^2)}{dt}} dW_t \qquad \gamma_t = 1$$
$$X_t \mid X_0 \sim \mathcal{N}\left(\gamma_t X_0, \sigma_t^2 I\right) \qquad \sigma_t^2 = \sigma_t^2$$

Although the mean is preserved, the variance explodes (if σ_t explodes).

Relative to the noise, the original signal X_0 is corrupted as $t \to \infty$.

Conditional distribution and variance calculations follow from (5.50) and (5.51) of S. Särkkä andA. Solin, *Applied Stochastic Differential Equations*, 2019.

General VP SDE

General variance preserving (VP) SDE:

$$dX_t = -\frac{\beta(t)}{2} X_t dt + \sqrt{\beta(t)} dW_t \qquad \gamma_t = e^{-\frac{1}{2}\int_0^t \beta(s)\, ds}$$

$$X_t \mid X_0 \sim \mathcal{N}\left(\gamma_t X_0, \sigma_t^2 I\right) \qquad \sigma_t^2 = 1 - e^{-\int_0^t \beta(s)\, ds}$$

In particular,

$$\mathrm{Var}(X_t) = I + e^{-\int_0^t \beta(s)\, ds}(\mathrm{Var}(X_0) - I)$$

and if $\mathrm{Var}(X_0) = I$, then

$$\mathrm{Var}(X_t) = I$$

So variance is “preserved”.

Conditional distribution and variance calculations follow from (5.50) and (5.51) of S. Särkkä andA. Solin, *Applied Stochastic Differential Equations*, 2019.

Training with O-U and DSM

Using $X_t \overset{\mathcal{D}}{=} \gamma_t X_0 + \sigma_t \varepsilon$, the score function simplifies to

$$\nabla_{X_t} \log p_t(X_t \mid X_0) = \frac{\gamma_t X_0 - X_t}{\sigma_t^2} \overset{\mathcal{D}}{=} -\frac{\varepsilon}{\sigma_t}$$

Define the *scaled score network*

$$\varepsilon_\theta(X_t, t) = -\sigma_t s_\theta(X_t, t)$$

Then the denoising score matching loss becomes

$$\begin{aligned}
\mathcal{L}(\theta) &= \int_0^T \lambda(t) \mathbb{E}_{X_0} \left[\mathbb{E}_{X_t \mid X_0} [\| s_\theta(X_t, t) - \nabla_{X_t} \log p_{t|0}(X_t \mid X_0) \|^2 \mid X_0] \right] dt \\
&= \int_0^T \frac{\lambda(t)}{\sigma_t^2} \mathbb{E}_{X_0} \left[\mathbb{E}_{\varepsilon \sim \mathcal{N}(0,I)} [\| \varepsilon_\theta(\gamma_t X_0 + \sigma_t \varepsilon, t) - \varepsilon \|^2 \mid X_0] \right] dt \\
&= T \mathop{\mathbb{E}}_{\substack{X_0 \sim p_0 \\ t \sim \mathrm{Uniform}([0,T]) \\ \varepsilon \sim \mathcal{N}(0,I)}} \left[\frac{\lambda(t)}{\sigma_t^2} \| \varepsilon_\theta(\gamma_t X_0 + \sigma_t \varepsilon, t) - \varepsilon \|^2 \right]
\end{aligned}$$

Interpretation of scaled score network

$\varepsilon_\theta(X_t, t)$ predicts noise ε from noised data $X_t \overset{\mathcal{D}}{=} \gamma_t X_0 + \sigma_t \varepsilon$.

Usually, the deep neural network represents ε_θ rather than s_θ. (Empirically works better.)

To clarify, ε_θ and s_θ only differ by a t-dependent (and θ-, data-independent) scaling factor.

Training with O-U and DSM

Using the Ornstein–Uhlenbeck forward SDE and the denoising score matching loss (i), we get the training routine:

while (not converged)

$X_0 \sim p_0 = p_{\text{data}}$

$t \sim \text{Uniform}([0, T])$

$\varepsilon \sim \mathcal{N}(0, I)$

$X_t = \gamma_t X_0 + \sigma_t \varepsilon$

Call optimizer with $\frac{\lambda(t)}{\sigma_t^2} \nabla_\theta \|\varepsilon_\theta(X_t, t) - \varepsilon\|^2$

end

Blow-up at $t = 0$

For both VP and VE SDEs, $\sigma_0 = 0$ and the loss blows up. Several ways to deal with this.

Option 1: Start the integral from a small $\delta > 0$

$$\mathcal{L}(\theta) = \int_{\delta}^{T} \frac{\lambda(t)}{\sigma_t^2} \mathop{\mathbb{E}}_{\substack{X_0 \sim p_0 \\ \varepsilon \sim \mathcal{N}(0,I)}} \left\| \varepsilon_\theta(\gamma_t X_0 - \sigma_t \varepsilon, t) - \varepsilon \right\|^2 dt$$

Option 2: Choose $\lambda(t) \to 0$ as $t \to 0$ so that $\lambda(t)/\sigma_t^2$ does not blow up. This makes the mean well-behaved, but the variance of the stochastic gradients may still blow up as $t \to 0$.

Option 3: Use importance sampling to reduce the variance as $t \to 0$.

Training with SSM

Using the sliced score matching loss (ii)

$$\mathcal{L}(\theta) = \int_0^T \lambda(t)\mathbb{E}_{X_t}\left[\|s_\theta(X_t,t)\|^2 + 2\mathbb{E}_\nu\left[\frac{d}{dh}\nu^\intercal s_\theta(X_t+h\nu,t)\Big|_{h=0}\right]\right]\,dt$$

we get the training routine:

while (not converged)

$t \sim \text{Uniform}([0,T])$

$X_t \sim p_t$ #forward-simulate SDE from $X_0 \sim p_{\text{data}}$

$\nu \sim p_\nu$ #$\mathbb{E}_{\nu\sim p_\nu}[\nu\nu^\top] = I$

Backprop on h with $\frac{d}{dh}\nu^\intercal s_\theta(X_t+h\nu,t)\Big|_{h=0}$

Call optimizer with $\lambda(t)\nabla_\theta\left(\|s_\theta(X_t,t)\|^2 + 2\frac{d}{dh}\nu^\intercal s_\theta(X_t+h\nu,t)\Big|_{h=0}\right)$

end

DSM vs SSM

SSM is more broadly applicable than DSM.

- SSM requires efficient sampling of X_t given X_0.
- DSM additionally requires evaluation of conditional density $p_{t|0}(X_t|X_0)$.
  (More precisely, the conditional score $\nabla_{X_T} \log p_{t|0}(X_t|X_0)$ is required.)

SSM allows a broader range of forward-diffusions to be used. Useful in, say, DSB.[#]

When applicable, DSM performs better than SSM.

SSM requires mixed (2nd-order) derivatives, while DSM requires 1st-order derivatives.
(Most modern DL libraries are capable of efficiently computing higher-order derivatives.)

[#]V. De Bortoli, J. Thornton, J. Heng, and A. Doucet, Diffusion Schrödinger bridge with applications to score-based generative modeling, *NeurIPS*, 2021.

SDE Sampling with trained score

Once s_θ has been trained, we can generate new samples with the approximate the reserve-time SDE

$$d\overline{X}_t = (f(\overline{X}_t, t) - g^2(t) s_\theta(\overline{X}_t, t))dt + g(t) d\overline{W}_t, \qquad \overline{X}_T \sim \mathcal{N}(0, \sigma_T^2 I)$$

Usually, one uses the reverse-time Ornstein–Uhlenbeck process

$$\begin{aligned} d\overline{X}_t &= (-\beta \overline{X}_t - \sigma^2 s_\theta(\overline{X}_t, t))dt + \sigma d\bar{W}_t, \qquad \overline{X}_T \sim \mathcal{N}(0, \sigma_T^2 I) \\ &= \left(\frac{\sigma^2}{\sigma_t} \varepsilon_\theta(\overline{X}_t, t) - \beta \overline{X}_t \right) dt + \sigma d\bar{W}_t \end{aligned}$$

SDE Sampling with trained score

Using a standard discretization (Euler–Maruyama), we get

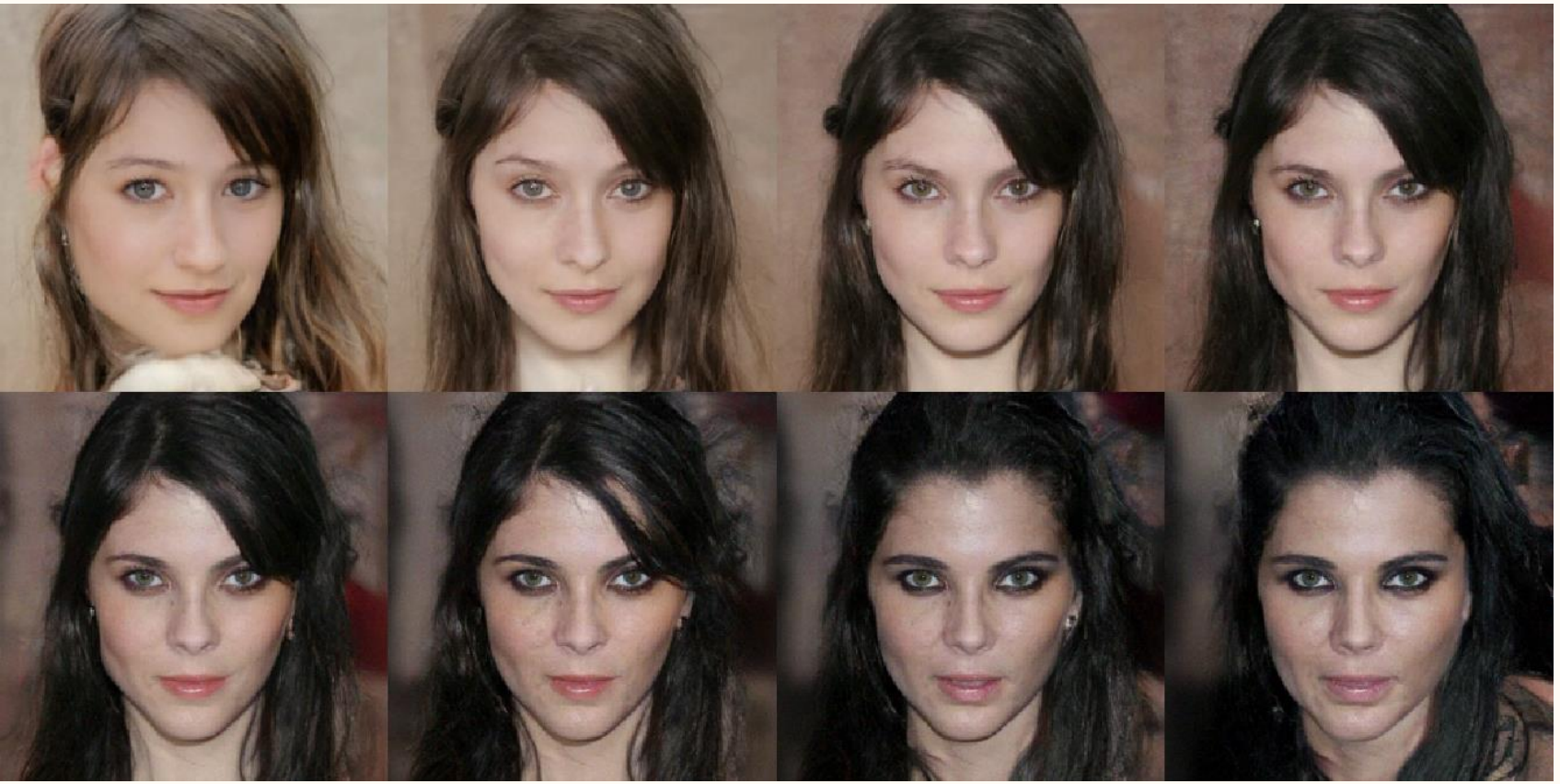
$$
\begin{aligned}
&\overline{X}_K \sim \mathcal{N}(0, \sigma_T^2 I) \\
&\text{for } k = K, K-1, \ldots, 2, 1 \\
&\quad Z_k \sim \mathcal{N}(0, I) \\
&\quad \overline{X}_{k-1} = \overline{X}_k - \Delta t \left(\frac{\sigma^2}{\sigma_t} \varepsilon_\theta(\overline{X}_k, k\Delta t) - \beta \overline{X}_k \right) + \sigma \sqrt{\Delta t} Z_k \\
&\text{end}
\end{aligned}
$$

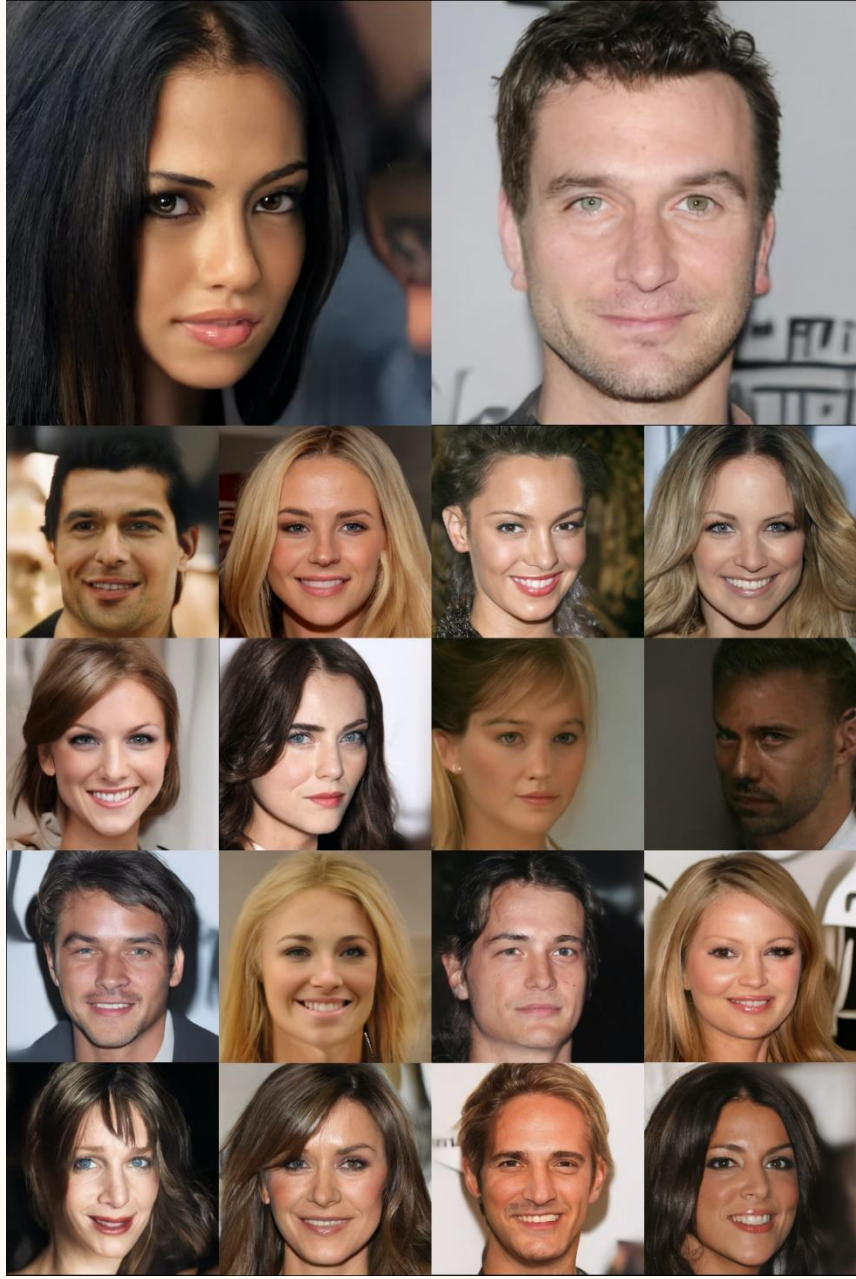
Output is $\bar{X}_0$ approximately distributed as p_0.

Called *DDPM sampling* for reasons to be explained later.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, Score-based generative modeling through stochastic differential equations, *ICLR*, 2021.

Samples via SDE





Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, Score-based generative modeling through stochastic differential equations, *ICLR*, 2021.

ODE Sampling with trained score

Once s_θ has been trained, we can also use approximate reserve-time ODE

$$d\overline{X}_t = \left(f(\overline{X}_t, t) - \frac{g^2(t)}{2} s_\theta(\overline{X}_t, t) \right) dt, \qquad \overline{X}_T \sim \mathcal{N}(0, \sigma_T^2 I)$$

Usually, one uses the reverse-time ODE of Ornstein–Uhlenbeck process

$$\begin{aligned} d\overline{X}_t &= \left(-\beta \overline{X}_t - \frac{\sigma^2}{2} s_\theta(\overline{X}_t, t) \right) dt, \qquad \overline{X}_T \sim \mathcal{N}(0, \sigma_T^2 I) \\ &= \left(\frac{\sigma^2}{2\sigma_t} \varepsilon_\theta(\overline{X}_t, t) - \beta \overline{X}_t \right) dt \end{aligned}$$

ODE Sampling with trained score

Using a standard discretization (Euler), we get

$$\overline{X}_K \sim \mathcal{N}(0, \sigma_T^2 I)$$
$$\text{for } k = K, K-1, \ldots, 2, 1$$
$$\overline{X}_{k-1} = \overline{X}_k - \Delta t \left(\frac{\sigma^2}{2\sigma_t} \varepsilon_\theta(\overline{X}_k, k\Delta t) - \beta \overline{X}_k \right)$$
$$\text{end}$$

Output is $\bar{X}_0$ approximately distributed as p_0.

This is called *DDIM sampling* for reasons to be explained later.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, Score-based generative modeling through stochastic differential equations, *ICLR*, 2021.

SDE vs ODE sampling

There is a more general family of SDEs that include standard SDE ($\lambda = 0$) and ODE ($\lambda = 1$) sampling. (Only $\lambda = 0$ and $\lambda = 1$ seems to be useful in practice.)

$$d\overline{X}_t = \left(f - \left(1 - \frac{\lambda}{2} \right) g^2 \nabla_x \log p_t \right) dt + \sqrt{1-\lambda} g d\bar{W}_t, \qquad \overline{X}_T \sim p_T$$

SDE sampling produces higher fidelity (based in visual inspection) images. Why? Theoretically, not understood well. Intuitively, noise steps of SDE sampling corrects for any errors from inaccurate terminal distribution p_T, inaccurate score function, and discretization.

However, ODE sampling is useful for applications such as image interpolation, which can be used for image editing (more on this later), and for likelihood computation (based on the observation that the ODE sampling defines a flow model).

(16) of Q. Zhang and Y. Chen, Diffusion normalizing flow, *NeurIPS*, 2021.
(27) C.-W. Huang, J. H. Lim, and A. Courville, A variational perspective on diffusion-based generative models and score matching, *NeurIPS*, 2021.

Image interpolation with ODE

Let $X^{(1)}$ and $X^{(2)}$ be images. Use the forward-time ODE

$$dX_t^{(i)} = \left(f(X_t^{(i)}, t) - \frac{g^2(t)}{2} s_\theta(X_t^{(i)}, t) \right) dt, \qquad X_0^{(i)} = X^{(i)},\ i = 1, 2$$

to obtain $X_T^{(1)}$ and $X_T^{(2)}$, which will look like pure noise sampled from $\mathcal{N}(0, \sigma_T^2)$. Form

$$X_T^\theta = \theta X_T^{(1)} + (1 - \theta) X_T^{(2)}$$

for $\theta \in [0,1]$. Use the reverse-time ODE

$$dX_t^\theta = \left(f(X_t^\theta, t) - \frac{g^2(t)}{2} s_\theta(X_t^\theta, t) \right) dt, \qquad \text{given } X_T^\theta$$

to obtain X_0^θ. This image will be a semantically meaningful interpolation of $X^{(1)}$ and $X^{(2)}$.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, Score-based generative modeling through stochastic differential equations, *ICLR*, 2021.

Why?

Why does this work?

If $X^{(1)}, X^{(2)} \sim p_0$, then $X^{(1)}, X^{(2)}$ will have high likelihood under p_0. Then $X_T^{(1)}, X_T^{(2)}$ will look like samples from $\mathcal{N}(0, \sigma_T^2)$, i.e., $X_T^{(1)}, X_T^{(2)}$ will have high likelihood under $\mathcal{N}(0, \sigma_T^2)$. Since $\mathcal{N}(0, \sigma_T^2)$ is a log-concave distribution, the interpolant X_T^θ will also have high likelihood. So its corresponding X_0^θ will be a realistic image with high likelihood under p_0.