Diffusion models

SDE-based perspectives

Julius Berner

October 12, 2022

University of Vienna

Introduction

Sampling from high-dimensional distributions

Task

Sample from a high-dimensional distribution Y_0.

Sampling from high-dimensional distributions

Task

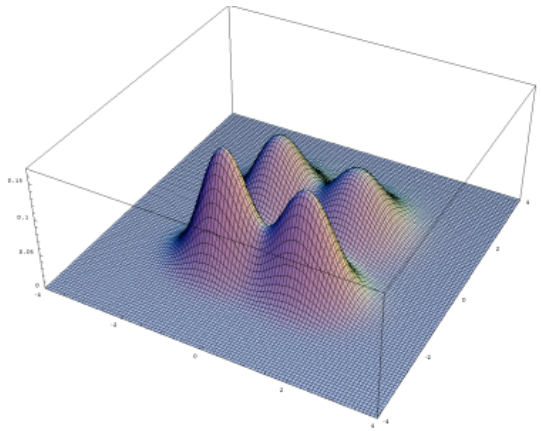
Sample from a high-dimensional distribution Y_0.

Y_0 can be given in the form of:

1. **samples** $Y_0^{(i)}$ Y_0 (images, text, sound, ...).

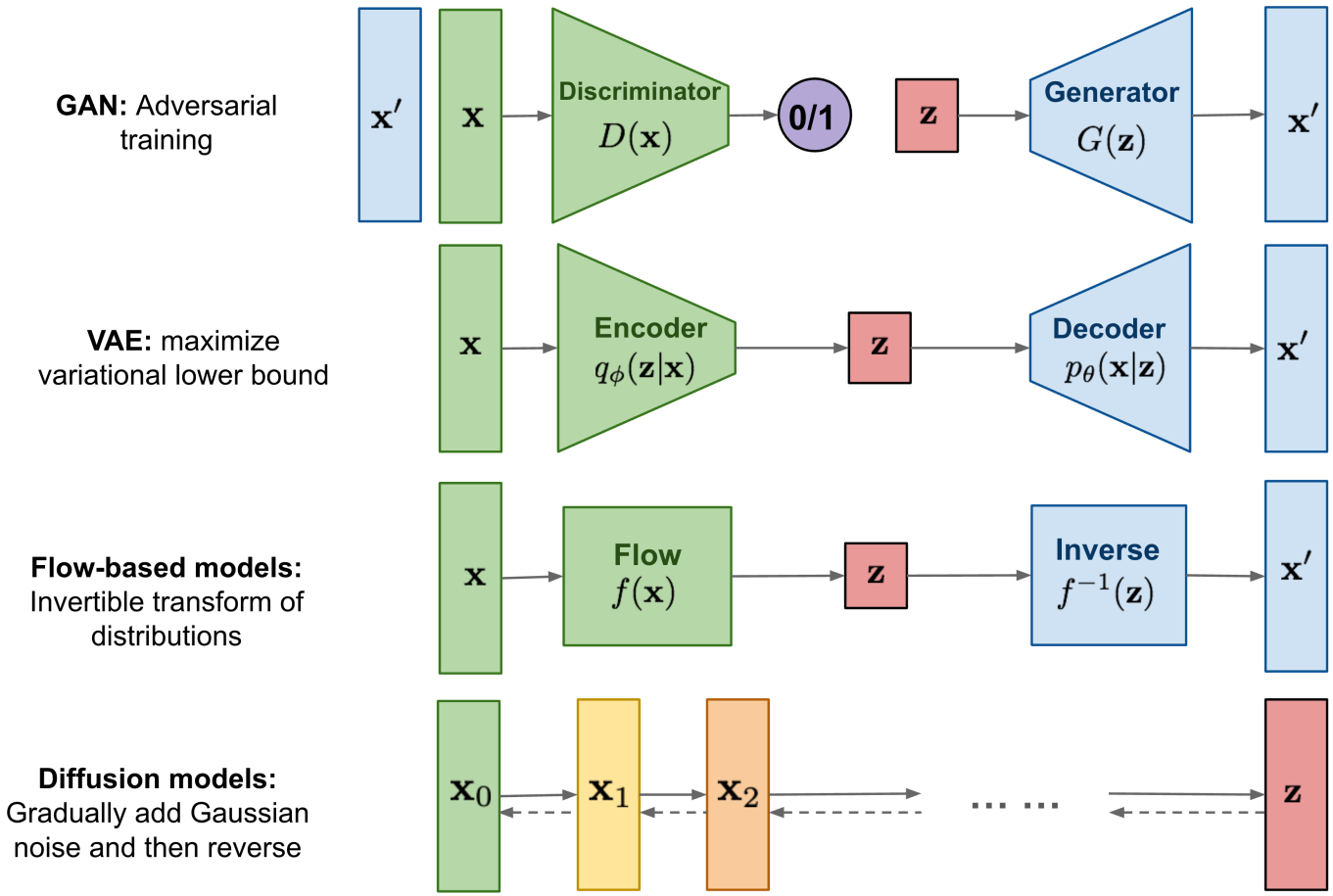
https://en.m.wikipedia.org/wiki/File:Cat_poster_1.jpg

2. an (unnormalized) **density** with $p_{Y_0} =$ $=Z$ (e.g., in Bayesian statistics, computational physics and chemistry).



https://en.wikipedia.org/wiki/File:Bimodal-bivariate-small.png

Overview of generative models



https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

History: The development of diffusion models builds upon *(denoising) di usion probabilistic modeling* [Sohl-Dickstein et al., 2015, Ho et al., 2020] and *score matching with Langevin dynamics* [Song and Ermon, 2019].

Diffusion models

State-of-the art in **generative modeling and likelihood estimation** of high-dimensional image data [Nichol and Dhariwal, 2021, Kingma et al., 2021].

Diffusion models

State-of-the art in **generative modeling and likelihood estimation** of high-dimensional image data [Nichol and Dhariwal, 2021, Kingma et al., 2021].

Figure 1: Sampling conditioned on the text prompt "a photograph of an astronaut riding a horse" using the stable diffusion model [Rombach et al., 2021].

Engineering Perspective

Diffusion process

Diffusion process Y_t: Gradually add coordinate-wise Gaussian noise, i.e., conditioned on d-dimensional data Y_0, we have that

$$Y_t = \quad _t Y_0 + \quad _t N; \quad N \quad N(0; I); \quad t \; 2 \; [0; T]:$$

[Nichol and Dhariwal, 2021]

Diffusion process

Diffusion process Y_t: Gradually add coordinate-wise Gaussian noise, i.e., conditioned on d-dimensional data Y_0, we have that

$$Y_t = \quad _t Y_0 + \quad _t N; \quad N \quad N(0; \mathrm{I}); \quad t \ 2 \ [0; T]:$$

[Nichol and Dhariwal, 2021]

Typical noise schedules for $\quad _t$ and $\quad _t = \quad \mathrm{q}\,\overline{\quad\quad}$:

$$1 \quad \quad ^2_t$$

Training

Noise prediction objective (with batch-size n):

$$L(\ ) = \sum_{i=1}^{n} \quad N^{(i)} \quad \Phi\ (Y_t^{(i)}; t^{(i)})\ \ ^2;$$

where Φ is typically a U-Net (with sinusoidal positional embeddings for t) and

- $Y_t^{(i)} = \ _{t^{(i)}} Y_0^{(i)} + \ _{t^{(i)}} N^{(i)}$ (noisy image)
- $N^{(i)} \quad N(0; \mathrm{I})$ (standardized noise)
- $t^{(i)} \quad U([0; T])$ (time)
- $Y^{(i)} \quad Y_0$ (data)

are i.i.d. samples.

Training

Noise prediction objective (with batch-size n):

$$L(\ ) = \sum_{i=1}^{n} \quad N^{(i)} \quad \Phi \ (Y_t^{(i)}; t^{(i)}) \ \ ^2;$$

where Φ is typically a U-Net (with sinusoidal positional embeddings for t) and

- $Y_t^{(i)} = \ _{t^{(i)}} Y_0^{(i)} + \ _{t^{(i)}} N^{(i)}$ (noisy image)
- $N^{(i)} \quad N(0; \mathrm{I})$ (standardized noise)
- $t^{(i)} \quad U([0; T])$ (time)
- $Y^{(i)} \quad Y_0$ (data)

are i.i.d. samples.

This is a **reparametrization of a denoising objective**, which works better in practice.

After training, we can approximately denoise Y_t as follows:

$$Y_0 \quad \frac{Y_t \quad \ _t \Phi \ (Y_t; t)}{t}.$$

Architecture of typical U-Nets

Architecture of typical U-Nets

https://www.assemblyai.com/blog/how-imagen-actually-works/

Sampling

Bayes' theorem yields the following formula for Y_s (conditioned on Y_0 and Y_t with $s < t$):

$$Y_s = \Theta_{t;s}(Y_0; Y_t; N); \quad N \quad N(0; I);$$

where

$$\Theta_{t;s}(Y_0; Y_t; N) = \underbrace{\frac{s^2}{t^2} Y_t + s \frac{t^2 - s^2}{t^2} Y_0}_{\text{mean}} + \underbrace{s \sqrt{\frac{t^2 - s^2}{t^2}}}_{\text{standard deviation}} N:$$

Sampling

Bayes' theorem yields the following formula for Y_s (conditioned on Y_0 and Y_t with $s < t$):

$$Y_s = \Theta_{t;s}(Y_0; Y_t; N); \quad N \quad N(0; I);$$

where

$$\Theta_{t;s}(Y_0; Y_t; N) = \underbrace{\frac{\frac{2}{s}\ \frac{t}{}}{\frac{2}{t}\ \frac{}{s}} Y_t + \frac{s}{\frac{2}{s}\ \frac{2}{t}} Y_0}_{\text{mean}} + \underbrace{\frac{s}{\frac{2}{s}\ \frac{4}{s}\ \frac{2}{t}\ \frac{2}{t}} N}_{\text{standard deviation}}:$$

Idea: Use the NN prediction for Y_0 and perform ancestral sampling.

1. Sample $X_T \quad N(0; I)$ (approximately distributed as Y_T).
2. Iterate:

$$X_{t\ \ 1} := \Theta_{t;t\ \ 1}\left(\underbrace{\frac{X_t \quad {}_t\Phi\ (X_t; t)}{\frac{t}{}}}_{\textit{denoising}}; X_t; N^{(t)}\right)$$

   with i.i.d. $N^{(t)} \quad N(0; I)$.
3. Output X_0 (approximately distributed as the data Y_0).

[Ho et al., 2020]

Sampling

Bayes' theorem yields the following formula for Y_s (conditioned on Y_0 and Y_t with $s < t$):

$$Y_s = \Theta_{t;s}(Y_0; Y_t; N); \quad N \quad N(0; I);$$

where

$$\Theta_{t;s}(Y_0; Y_t; N) = \underbrace{\frac{\frac{2}{s}\ \frac{t}{}}{\frac{2}{t}\ \frac{s}{}} Y_t + \frac{s}{\frac{2}{s}\ \frac{2}{t}} Y_0}_{\text{mean}} + \underbrace{\frac{s}{\frac{2}{s}\ \frac{2}{t}}\ \frac{\frac{2}{t}\ \frac{4}{s}}{\frac{2}{s}\ \frac{2}{t}} N}_{\text{standard deviation}}:$$

Idea: Use the NN prediction for Y_0 and perform ancestral sampling.

1. Sample $X_T \quad N(0; I)$ (approximately distributed as Y_T).
2. Iterate:

$$X_{t\ \ 1} := \Theta_{t;t\ \ 1}\left(\underbrace{\frac{X_t \quad {}_t\Phi\ (X_t; t)}{\frac{t}{}}}_{\text{denoising}}; X_t; N^{(t)}\right)$$

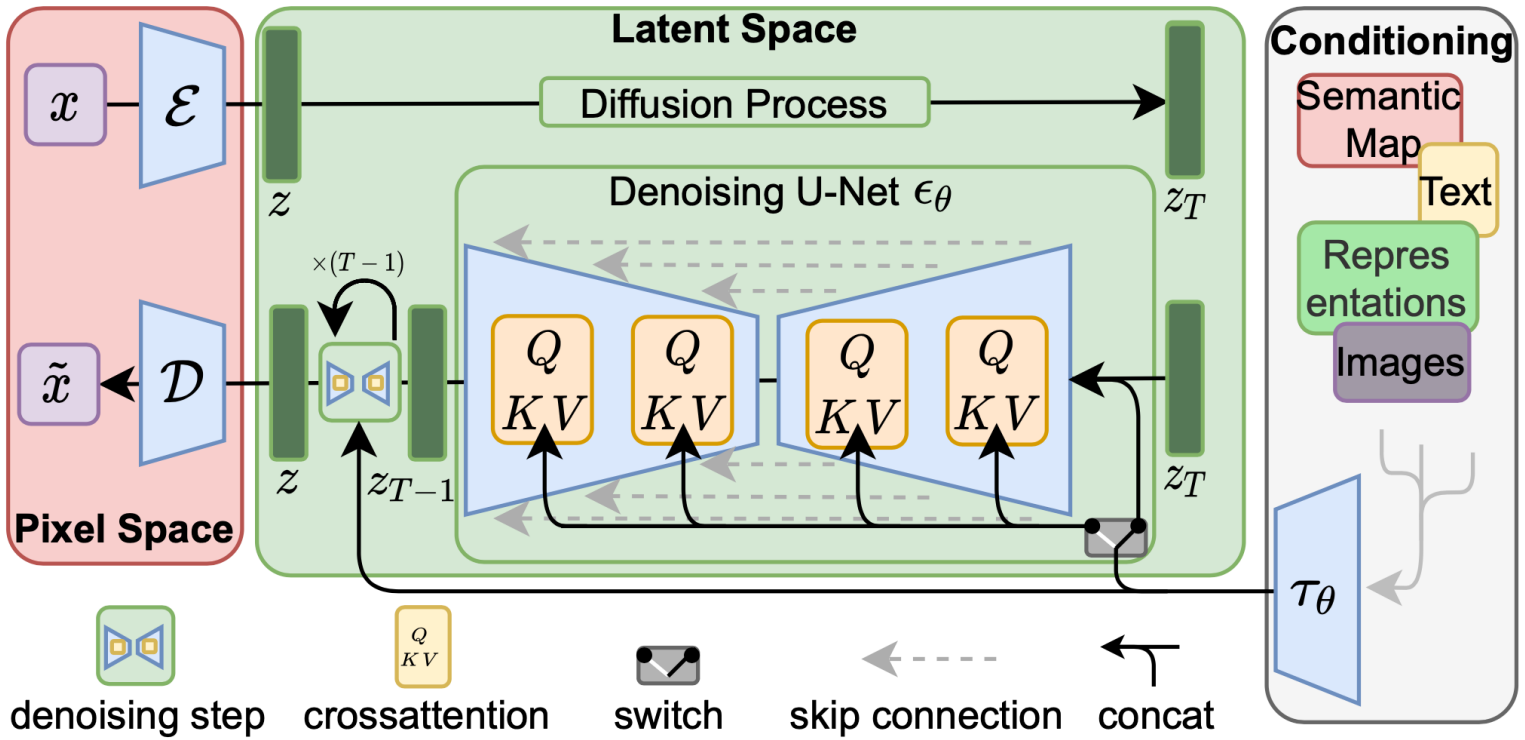
   with i.i.d. $N^{(t)} \quad N(0; I)$.
3. Output X_0 (approximately distributed as the data Y_0).

This can be viewed as variational auto-encoder with fixed encoder.

[Ho et al., 2020]

Stable diffusion model

Use diffusion in latent space of a pre-trained (regularized) auto-encoder and condition the U-Net on features given by a pre-trained domain-specific encoder (e.g., a transformer for text prompts):



[Rombach et al., 2021]

SDE-based perspective

Stochastic differential equations (SDEs)

Consider solutions Y to SDEs of the form

$$\mathrm{d}Y_s = \underbrace{(Y_s)}_{\text{drift}}\,\mathrm{d}s + \underbrace{(Y_s)}_{\text{diffusion}}\,\mathrm{d}B_s;$$

where B_s is a d-dimensional Brownian motion.

Stochastic differential equations (SDEs)

Consider solutions Y to SDEs of the form

$$\mathrm{d}Y_s = \underbrace{(Y_s)}_{\text{drift}}\,\mathrm{d}s + \underbrace{(Y_s)}_{\text{diffusion}}\,\mathrm{d}B_s;$$

where B_s is a d-dimensional Brownian motion.

Intuition via Euler-Maruyama scheme $\hat{Y}_{t_{k+1}}$ Y_{t_k}:

$$\hat{Y}_{t_{k+1}} = \hat{Y}_{t_k} + (\hat{Y}_{t_k})(t_{k+1} \quad t_k) + (\hat{Y}_{t_k})\underbrace{(B_{t_{k+1}} \quad B_{t_k})}_{N(0;t_{k+1} \quad t_k)}:$$

Figure 2: SDE solution and Euler-Maruyama scheme with $t_k = \frac{kT}{N}$ and $N = 4; 8$.

Inference SDE

Consider a (time-varying) Ornstein–Uhlenbeck process

$$\mathrm{d}Y_s = \quad (s)Y_s\mathrm{d}s + \quad (s)\mathrm{d}B_s;$$

which diffuses the data Y_0.

[Song et al., 2020]

Inference SDE

Consider a (time-varying) Ornstein–Uhlenbeck process

$$\mathrm{d}Y_s = \quad (s)Y_s\mathrm{d}s + \quad (s)\mathrm{d}B_s;$$

which diffuses the data Y_0.

[Song et al., 2020]

Note that, conditioned on Y_0, the solution Y_s is normally distributed. For the choices

$$(s) = \frac{^0(s)}{(s)} \quad \text{and} \quad ^2(s) = 2 \ (s) \ ^0(s) \quad 2\frac{^0(s) \ ^2(s)}{(s)}$$

we recover $p_{Y_s jY_0}(jY_0) = N(\ _s Y_0; \ _s^2 \mathrm{I})$.

Generative SDE

We can reverse the diffusion (proven via the Fokker-Planck equation):

Reverse-time generative SDE/ODE [Anderson, 1982, Song et al., 2020]

The solutions to the SDE

$$\mathrm{d}X_s = \quad ^{>} r \log p_{Y_T \ s} \quad (X_s; s)\mathrm{d}s + \ (s)\mathrm{d}B_s; \quad X_0 \quad Y_T;$$

and the ODE

$$\mathrm{d}X_s = \frac{1}{2} \quad ^{>} r \log p_{Y_T \ s} \quad (X_s; s)\mathrm{d}s; \quad X_0 \quad Y_T;$$

both satisfy that $X_s \quad Y_T \ _s$, where $p_{Y_T \ s}$ is the density of $Y_T \ _s$.

Generative SDE

We can reverse the diffusion (proven via the Fokker-Planck equation):

Reverse-time generative SDE/ODE [Anderson, 1982, Song et al., 2020]

The solutions to the SDE

$$\mathrm{d}X_s = \quad ^{>} r \log p_{Y_T \quad s} \qquad (X_s; s)\mathrm{d}s + \quad (s)\mathrm{d}B_s; \quad X_0 \quad Y_T;$$

and the ODE

$$\mathrm{d}X_s = \quad \frac{1}{2} \quad ^{>} r \log p_{Y_T \quad s} \qquad (X_s; s)\mathrm{d}s; \quad X_0 \quad Y_T;$$

both satisfy that $X_s \quad Y_T \quad s$, where $p_{Y_T \quad s}$ is the density of $Y_T \quad s$.

- We need an approximation to the *score* $r \log p_{Y_T \quad s}$.

Generative SDE

We can reverse the diffusion (proven via the Fokker-Planck equation):

Reverse-time generative SDE/ODE [Anderson, 1982, Song et al., 2020]

The solutions to the SDE

$$\mathrm{d}X_s = \quad ^{>} r \log p_{Y_T \quad s} \qquad (X_s; s)\mathrm{d}s + \quad (s)\mathrm{d}B_s; \quad X_0 \quad Y_T;$$

and the ODE

$$\mathrm{d}X_s = \quad \frac{1}{2} \quad ^{>} r \log p_{Y_T \quad s} \qquad (X_s; s)\mathrm{d}s; \quad X_0 \quad Y_T;$$

both satisfy that $X_s \quad Y_T \quad s$, where $p_{Y_T \quad s}$ is the density of $Y_T \quad s$.

- We need an approximation to the *score* $r \log p_{Y_T \quad s}$.

Using the noise prediction network Φ , we obtain that

$$r \log p_{Y_t j Y_0}(Y_t j Y_0) = \frac{Y_t \quad {}_t Y_0}{{}_t^2} \quad \frac{\Phi \ (Y_t; t)}{{}_t}.$$

Generative SDE

We can reverse the diffusion (proven via the Fokker-Planck equation):

Reverse-time generative SDE/ODE [Anderson, 1982, Song et al., 2020]

The solutions to the SDE

$$dX_s = \quad ^{>} r \log p_{Y_T \quad s} \qquad (X_s; s)ds + \quad (s)dB_s; \quad X_0 \quad Y_T;$$

and the ODE

$$dX_s = \frac{1}{2} \quad ^{>} r \log p_{Y_T \quad s} \qquad (X_s; s)ds; \quad X_0 \quad Y_T;$$

both satisfy that $X_s \quad Y_T \quad s$, where $p_{Y_T \quad s}$ is the density of $Y_T \quad s$.

- We need an approximation to the *score* $r \log p_{Y_T \quad s}$.

Using the noise prediction network Φ , we obtain that

$$r \log p_{Y_t j Y_0}(Y_t j Y_0) = \frac{Y_t \quad _t Y_0}{^2_t} \quad \frac{\Phi \ (Y_t; t)}{t}:$$

Sampling:

1. Sample $X_0 \quad N(0; I)$.
2. Plug-in the approximate score and simulate the SDE (using Euler-Maruyama) or the ODE (analogous to time-continuous normalizing flows) to obtain samples X_T.

Variational Lower Bound

Up to a constant and a time-dependent weighting, the (negative) **noise prediction objective** also provides a **lower bound on the log-likelihood** $\log p_{X_T}(Y_0)$ of our model X (with score replaced by the NN approximation).

Variational Lower Bound

Up to a constant and a time-dependent weighting, the (negative) **noise prediction objective** also provides a **lower bound on the log-likelihood** $\log p_{X_T}(Y_0)$ of our model X (with score replaced by the NN approximation).

Proof idea with short-hands $f(t) := f(T - t)$, $D = \frac{1}{2}$, and $X = X$:

1. Fokker-Planck for p_X:

$$\partial_t p_X = \operatorname{div}(\operatorname{div}(D \nabla p_X) - p_X)$$

Variational Lower Bound

Up to a constant and a time-dependent weighting, the (negative) **noise prediction objective** also provides a **lower bound on the log-likelihood** $\log p_{X_T}(Y_0)$ of our model X (with score replaced by the NN approximation).

Proof idea with short-hands $f(t) := f(T - t)$, $D = \frac{1}{2}$ $^{>}$, and $X = X$:

1. Fokker-Planck for p_X:

$$\partial_t p_X = \operatorname{div}\ \operatorname{div}\ D p_X - p_X$$

2. Kolmogorov backwards equation for p_X:

$$\partial_t p_X = \operatorname{tr}\ D\ r^2 p_X + r\, p_X + \operatorname{div}(\ ) p_X :$$

Variational Lower Bound

Up to a constant and a time-dependent weighting, the (negative) **noise prediction objective** also provides a **lower bound on the log-likelihood** $\log p_{X_T}(Y_0)$ of our model X (with score replaced by the NN approximation).

Proof idea with short-hands $f(t) := f(T - t)$, $D = \frac{1}{2}\sigma\sigma^\top$, and $X = X^\theta$:

1. Fokker-Planck for p_X:
$$\partial_t p_X = \operatorname{div}(\operatorname{div}(D p_X) - \mu p_X)$$

2. Kolmogorov backwards equation for p_X:
$$\partial_t p_X = \operatorname{tr}(D \nabla^2 p_X) + \mu \cdot \nabla p_X + \operatorname{div}(\mu) p_X.$$

3. HJB equation for $V := \log p_X$ (Hopf–Cole transformation):
$$\partial_t V = \operatorname{tr}(D \nabla^2 V) + \mu \cdot \nabla V - \operatorname{div}(\mu) + \tfrac{1}{2}\sigma\sigma^\top \nabla V^2; \quad V(\cdot, T) = \log p_{X_0}.$$

Variational Lower Bound

Up to a constant and a time-dependent weighting, the (negative) **noise prediction objective** also provides a **lower bound on the log-likelihood** $\log p_{X_T}(Y_0)$ of our model X (with score replaced by the NN approximation).

Proof idea with short-hands $f(t) := f(T - t)$, $D = \frac{1}{2}$ $^{>}$, and $X = X$:

1. Fokker-Planck for p_X:

$$\partial_t p_X = \operatorname{div} \operatorname{div} D p_X - p_X$$

2. Kolmogorov backwards equation for p_X:

$$\partial_t p_X = \operatorname{tr} D \nabla^2 p_X + \nabla p_X + \operatorname{div}( ) p_X :$$

3. HJB equation for $V := \log p_X$ (Hopf–Cole transformation):

$$\partial_t V = \operatorname{tr} D \nabla^2 V + \nabla V \operatorname{div}( ) + \frac{1}{2} \nabla V^2; \quad V(; T) = \log p_{X_0} :$$

4. Reparametrize and use verification theorem from optimal control:

$$\left[\log p_{X_T}(Y_0) \right] \int_0^T \operatorname{div}(\Phi) - \frac{1}{2} k\Phi k^2 (Y_s; s) \, ds + \log p_{X_0}(Y_T) :$$

Variational Lower Bound

Up to a constant and a time-dependent weighting, the (negative) **noise prediction objective** also provides a **lower bound on the log-likelihood** $\log p_{X_T}(Y_0)$ of our model X (with score replaced by the NN approximation).

Proof idea with short-hands $f(t) := f(T - t)$, $D = \frac{1}{2}$ ^{2}, and $X = X$:

1. Fokker-Planck for p_X:
$$\partial_t p_X = \operatorname{div}\ \operatorname{div}\ D p_X - p_X$$

2. Kolmogorov backwards equation for p_X:
$$\partial_t p_X = \operatorname{tr}\ D\ r^2 p_X + r p_X + \operatorname{div}(\ ) p_X :$$

3. HJB equation for $V := \log p_X$ (Hopf–Cole transformation):
$$\partial_t V = \operatorname{tr}\ D\ r^2 V + r V \operatorname{div}(\ ) + \frac{1}{2}\ ^2 r V\ ^2; \quad V(\ ; T) = \log p_{X_0} :$$

4. Reparametrize and use verification theorem from optimal control:
$$\left[\log p_{X_T}(Y_0) \right]_0^T \operatorname{div}(\ \Phi\ ) \frac{1}{2} k\Phi\ k^2\ (Y_s; s)\, \mathrm{d}s + \log p_{X_0}(Y_T) :$$

5. Employ Stokes' theorem to rewrite the divergence.

References i

Anderson, B. D. (1982).
Reverse-time diffusion equation models.
Stochastic Processes and their Applications, 12(3):313–326.

Ho, J., Jain, A., and Abbeel, P. (2020).
Denoising diffusion probabilistic models.
Advances in Neural Information Processing Systems, 33:6840–6851.

Kingma, D., Salimans, T., Poole, B., and Ho, J. (2021).
Variational diffusion models.
Advances in Neural Information Processing Systems, 34:21696–21707.

Nichol, A. Q. and Dhariwal, P. (2021).
Improved denoising diffusion probabilistic models.
In *International Conference on Machine Learning*, pages 8162–8171. PMLR.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2021).
High-resolution image synthesis with latent diffusion models.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015).
Deep unsupervised learning using nonequilibrium thermodynamics.
In *International Conference on Machine Learning*, pages 2256–2265. PMLR.

Song, Y. and Ermon, S. (2019).
Generative modeling by estimating gradients of the data distribution.
Advances in Neural Information Processing Systems, 32.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2020).
Score-based generative modeling through stochastic differential equations.
In *International Conference on Learning Representations*.