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

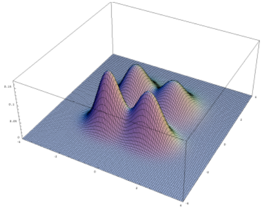
Sample from a high-dimensional distribution Y_0.

Y_0 can be given in the form of:

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

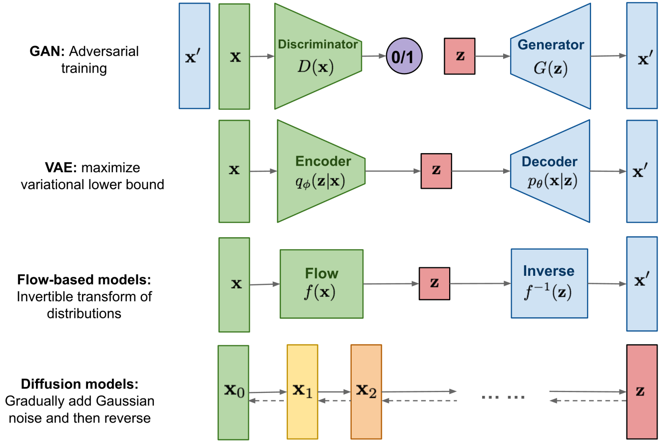
https://en.m.wikipedia.org/wiki/File:Cat_poster_1.jpg

2. an (unnormalized) **density** ρ with $p_{Y_0} = \rho/\mathcal{Z}$ (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) diffusion 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].

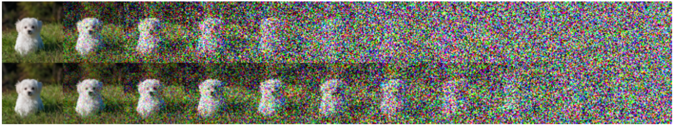
Figure 1: Sampling conditioned on the text prompt “a photograph of an astronaut riding a horse” using the stable diffusion model [Rombach et al., 2021].

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

$$Y_t = \alpha_t Y_0 + \beta_t N, \quad N \sim \mathcal{N}(0, \mathrm{I}), \quad t \in [0, T].$$

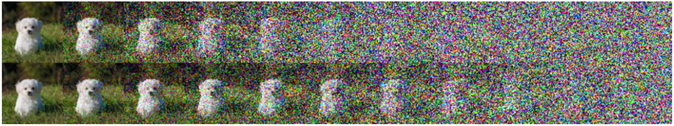


[Nichol and Dhariwal, 2021]

Diffusion process

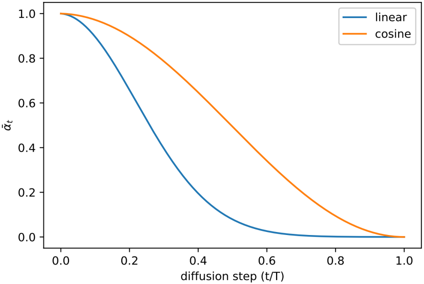
Diffusion process Y_t: Gradually add coordinate-wise Gaussian noise, i.e., conditioned on d-dimensional data Y_0, we have that

$$Y_t = \alpha_t Y_0 + \beta_t N, \quad N \sim \mathcal{N}(0, \mathrm{I}), \quad t \in [0, T].$$



[Nichol and Dhariwal, 2021]

Typical noise schedules for α_t and $\beta_t = \sqrt{1 - \alpha_t^2}$:



[Nichol and Dhariwal, 2021]

Training

Noise prediction objective (with batch-size n):

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} \left\| N^{(i)} - \Phi_\theta(Y_t^{(i)}, t^{(i)}) \right\|^2,$$

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

- $Y_t^{(i)} = \alpha_{t^{(i)}} Y_0^{(i)} + \beta_{t^{(i)}} N^{(i)}$ (noisy image)
- $N^{(i)} \sim \mathcal{N}(0, \mathrm{I})$ (standardized noise)
- $t^{(i)} \sim \mathcal{U}([0, T])$ (time)
- $Y^{(i)} \sim Y_0$ (data)

are i.i.d. samples.

Training

Noise prediction objective (with batch-size n):

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} \left\| N^{(i)} - \Phi_\theta(Y_t^{(i)}, t^{(i)}) \right\|^2,$$

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

- $Y_t^{(i)} = \alpha_{t^{(i)}} Y_0^{(i)} + \beta_{t^{(i)}} N^{(i)}$ (noisy image)
- $N^{(i)} \sim \mathcal{N}(0, \mathrm{I})$ (standardized noise)
- $t^{(i)} \sim \mathcal{U}([0, T])$ (time)
- $Y^{(i)} \sim Y_0$ (data)

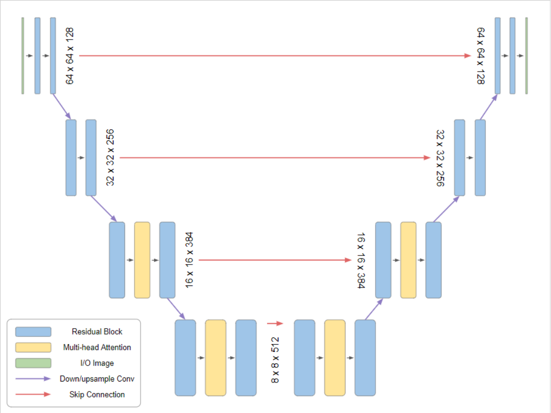
are i.i.d. samples.

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

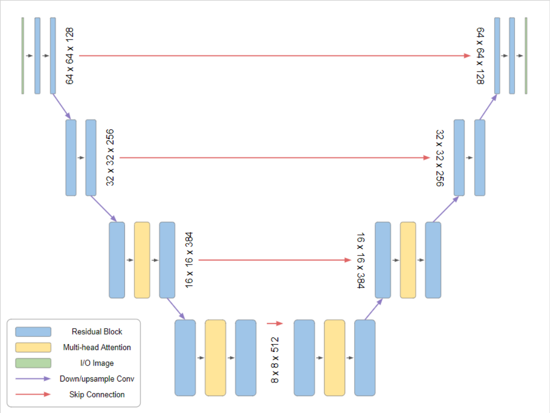
After training, we can approximately denoise Y_t as follows:

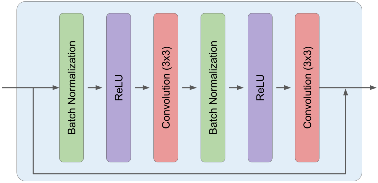
$$Y_0 \approx \frac{Y_t - \beta_t \Phi_\theta(Y_t, t)}{\alpha_t}.$$

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 $s < t$):

$$Y_s = \Theta_{t,s}(Y_0, Y_t, N), \quad N \sim \mathcal{N}(0, \mathrm{I}),$$

where

$$\Theta_{t,s}(Y_0, Y_t, N) = \underbrace{\frac{\beta_s^2 \alpha_t}{\beta_t^2 \alpha_s} Y_t + \left(\alpha_s - \frac{\alpha_t^2 \beta_s^2}{\alpha_s \beta_t^2} \right) Y_0}_{\text{mean}} + \underbrace{\sqrt{\beta_s^2 - \frac{\alpha_t^2 \beta_s^4}{\alpha_s^2 \beta_t^2}}}_{\text{standard deviation}} N.$$

Sampling

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

$$Y_s = \Theta_{t,s}(Y_0, Y_t, N), \quad N \sim \mathcal{N}(0, \mathrm{I}),$$

where

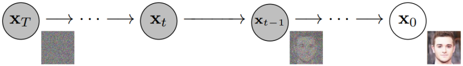
$$\Theta_{t,s}(Y_0, Y_t, N) = \underbrace{\frac{\beta_s^2 \alpha_t}{\beta_t^2 \alpha_s} Y_t + \left(\alpha_s - \frac{\alpha_t^2 \beta_s^2}{\alpha_s \beta_t^2}\right) Y_0}_{\text{mean}} + \underbrace{\sqrt{\beta_s^2 - \frac{\alpha_t^2 \beta_s^4}{\alpha_s^2 \beta_t^2}}}_{\text{standard deviation}} N.$$

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

1. Sample $X_T \sim \mathcal{N}(0, \mathrm{I})$ (approximately distributed as Y_T).
2. Iterate:

$$X_{t-1} := \Theta_{t,t-1}\left(\underbrace{\frac{X_t - \beta_t \Phi_\theta(X_t, t)}{\alpha_t}}_{\textit{denoising}}, X_t, N^{(t)}\right)$$

   with i.i.d. $N^{(t)} \sim \mathcal{N}(0, \mathrm{I})$.
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 $s < t$):

$$Y_s = \Theta_{t,s}(Y_0, Y_t, N), \quad N \sim \mathcal{N}(0, \mathrm{I}),$$

where

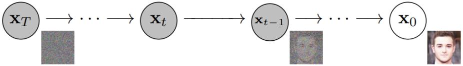
$$\Theta_{t,s}(Y_0, Y_t, N) = \underbrace{\frac{\beta_s^2 \alpha_t}{\beta_t^2 \alpha_s} Y_t + \left(\alpha_s - \frac{\alpha_t^2 \beta_s^2}{\alpha_s \beta_t^2} \right) Y_0}_{\text{mean}} + \underbrace{\sqrt{\beta_s^2 - \frac{\alpha_t^2 \beta_s^4}{\alpha_s^2 \beta_t^2}}}_{\text{standard deviation}} N.$$

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

1. Sample $X_T \sim \mathcal{N}(0, \mathrm{I})$ (approximately distributed as Y_T).
2. Iterate:

$$X_{t-1} := \Theta_{t,t-1}\left(\underbrace{\frac{X_t - \beta_t \Phi_\theta(X_t, t)}{\alpha_t}}_{\textit{denoising}}, X_t, N^{(t)} \right)$$

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

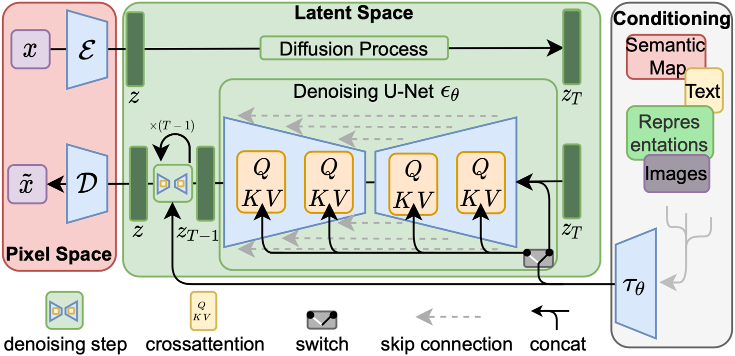


[Ho et al., 2020]

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

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

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

where B_s is a d-dimensional Brownian motion.

Stochastic differential equations (SDEs)

Consider solutions Y to SDEs of the form

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

where B_s is a d-dimensional Brownian motion.

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

$$\hat{Y}_{t_{k+1}} = \hat{Y}_{t_k} + \mu(\hat{Y}_{t_k})(t_{k+1} - t_k) + \sigma(\hat{Y}_{t_k})\underbrace{(B_{t_{k+1}} - B_{t_k})}_{\sim\mathcal{N}(0,t_{k+1}-t_k)}.$$

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

$$\mathrm{d}Y_s = \mu(s) Y_s \mathrm{d}s + \sigma(s) \mathrm{d}B_s,$$

which diffuses the data Y_0.

[Song et al., 2020]

Inference SDE

Consider a (time-varying) Ornstein–Uhlenbeck process

$$\mathrm{d}Y_s = \mu(s) Y_s \mathrm{d}s + \sigma(s) \mathrm{d}B_s,$$

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

$$\mu(s) = \frac{\alpha'(s)}{\alpha(s)} \quad \text{and} \quad \sigma^2(s) = 2\beta(s)\beta'(s) - 2\frac{\alpha'(s)\beta^2(s)}{\alpha(s)}$$

we recover $p_{Y_s|Y_0}(\cdot|Y_0) = \mathcal{N}(\alpha_s Y_0, \beta_s^2 \mathrm{I})$.

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

$$\mathrm{d}X_s = \left(\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s + \sigma(s)\mathrm{d}B_s, \quad X_0 \sim Y_T,$$

and the ODE

$$\mathrm{d}X_s = \left(\frac{1}{2}\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s, \quad X_0 \sim Y_T,$$

both satisfy that $X_s \sim Y_{T-s}$, 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

$$\mathrm{d}X_s = \left(\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s + \sigma(s)\mathrm{d}B_s, \quad X_0 \sim Y_T,$$

and the ODE

$$\mathrm{d}X_s = \left(\frac{1}{2}\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s, \quad X_0 \sim Y_T,$$

both satisfy that $X_s \sim Y_{T-s}$, where $p_{Y_{T-s}}$ is the density of Y_{T-s}.

⚠ We need an approximation to the *score* $\nabla \log p_{Y_{T-s}}$.

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

$$\mathrm{d}X_s = \left(\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s + \sigma(s)\mathrm{d}B_s, \quad X_0 \sim Y_T,$$

and the ODE

$$\mathrm{d}X_s = \left(\frac{1}{2}\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s, \quad X_0 \sim Y_T,$$

both satisfy that $X_s \sim Y_{T-s}$, where $p_{Y_{T-s}}$ is the density of Y_{T-s}.

⚠ We need an approximation to the *score* $\nabla \log p_{Y_{T-s}}$.

Using the noise prediction network Φ_θ, we obtain that

$$\nabla \log p_{Y_t|Y_0}(Y_t|Y_0) = \frac{Y_t - \alpha_t Y_0}{\beta_t^2} \approx \frac{\Phi_\theta(Y_t, t)}{\beta_t}.$$

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

$$\mathrm{d}X_s = \left(\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s + \sigma(s)\mathrm{d}B_s, \quad X_0 \sim Y_T,$$

and the ODE

$$\mathrm{d}X_s = \left(\frac{1}{2}\sigma\sigma^\top \nabla \log p_{Y_{T-s}} - \mu\right)(X_s, s)\mathrm{d}s, \quad X_0 \sim Y_T,$$

both satisfy that $X_s \sim Y_{T-s}$, where $p_{Y_{T-s}}$ is the density of Y_{T-s}.

⚠ We need an approximation to the *score* $\nabla \log p_{Y_{T-s}}$.

Using the noise prediction network Φ_θ, we obtain that

$$\nabla \log p_{Y_t|Y_0}(Y_t|Y_0) = \frac{Y_t - \alpha_t Y_0}{\beta_t^2} \approx \frac{\Phi_\theta(Y_t, t)}{\beta_t}.$$

Sampling:

1. Sample $X_0 \sim \mathcal{N}(0, \mathrm{I})$.
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** $\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right]$ of our model X^θ (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** $\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right]$ of our model X^θ (with score replaced by the NN approximation).

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

1. Fokker-Planck for p_X:

$$\partial_t p_X = \operatorname{div}\left(\operatorname{div}\left(\overleftarrow{D} p_X\right) - \overleftarrow{\mu} p_X\right)$$

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** $\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right]$ of our model X^θ (with score replaced by the NN approximation).

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

1. Fokker-Planck for p_X:

$$\partial_t p_X = \operatorname{div}\left(\operatorname{div}\left(\overleftarrow{D} p_X\right) - \overleftarrow{\mu} p_X\right)$$

2. Kolmogorov backwards equation for $\overleftarrow{p}_X$:

$$\partial_t \overleftarrow{p}_X = -\operatorname{tr}\left(D \nabla^2 \overleftarrow{p}_X\right) + \mu \cdot \nabla \overleftarrow{p}_X + \operatorname{div}(\mu) \overleftarrow{p}_X.$$

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** $\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right]$ of our model X^θ (with score replaced by the NN approximation).

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

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

2. Kolmogorov backwards equation for $\overleftarrow{p}_X$:
$$\partial_t \overleftarrow{p}_X = -\operatorname{tr}\left(D\nabla^2 \overleftarrow{p}_X\right) + \mu \cdot \nabla \overleftarrow{p}_X + \operatorname{div}(\mu)\overleftarrow{p}_X.$$

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

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** $\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right]$ of our model X^θ (with score replaced by the NN approximation).

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

1. Fokker-Planck for p_X:
$$\partial_t p_X = \operatorname{div}\left(\operatorname{div}\left(\overleftarrow{D} p_X\right) - \overleftarrow{\mu} p_X\right)$$
2. Kolmogorov backwards equation for $\overleftarrow{p}_X$:
$$\partial_t \overleftarrow{p}_X = -\operatorname{tr}\left(D\nabla^2 \overleftarrow{p}_X\right) + \mu \cdot \nabla \overleftarrow{p}_X + \operatorname{div}(\mu)\overleftarrow{p}_X.$$
3. HJB equation for $V := -\log \overleftarrow{p}_X$ (Hopf–Cole transformation):
$$\partial_t V = -\operatorname{tr}\left(D\nabla^2 V\right) + \mu \cdot \nabla V - \operatorname{div}(\mu) + \tfrac{1}{2}\left\|\sigma^\top \nabla V\right\|^2, \quad V(\cdot, T) = -\log p_{X_0}.$$
4. Reparametrize and use verification theorem from optimal control:
$$\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right] \geq \mathbb{E}\left[\int_0^T \left(-\operatorname{div}(\sigma\Phi_\theta - \mu) - \frac{1}{2}\|\Phi_\theta\|^2\right)(Y_s, s)\,\mathrm{d}s + \log p_{X_0^\theta}(Y_T)\right].$$

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** $\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right]$ of our model X^θ (with score replaced by the NN approximation).

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

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

2. Kolmogorov backwards equation for $\overleftarrow{p}_X$:
$$\partial_t \overleftarrow{p}_X = -\operatorname{tr}\left(D\nabla^2 \overleftarrow{p}_X\right) + \mu \cdot \nabla \overleftarrow{p}_X + \operatorname{div}(\mu)\overleftarrow{p}_X.$$

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

4. Reparametrize and use verification theorem from optimal control:
$$\mathbb{E}\left[\log p_{X_T^\theta}(Y_0)\right] \geq \mathbb{E}\left[\int_0^T \left(-\operatorname{div}(\sigma\Phi_\theta - \mu) - \frac{1}{2}\|\Phi_\theta\|^2\right)(Y_s, s)\,\mathrm{d}s + \log p_{X_0^\theta}(Y_T)\right].$$

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*.