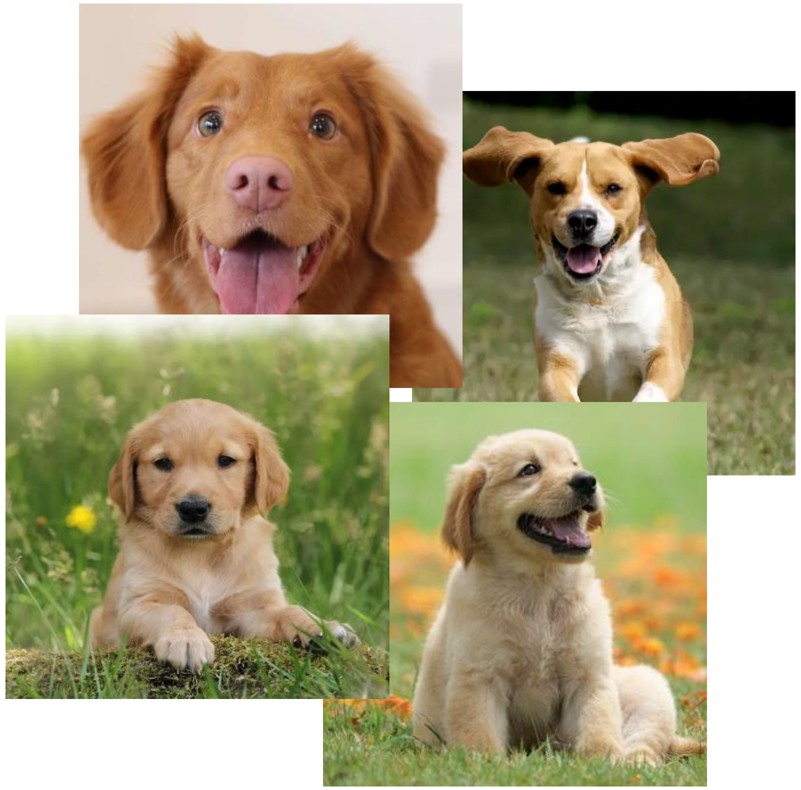
Diffusion Study Group #3

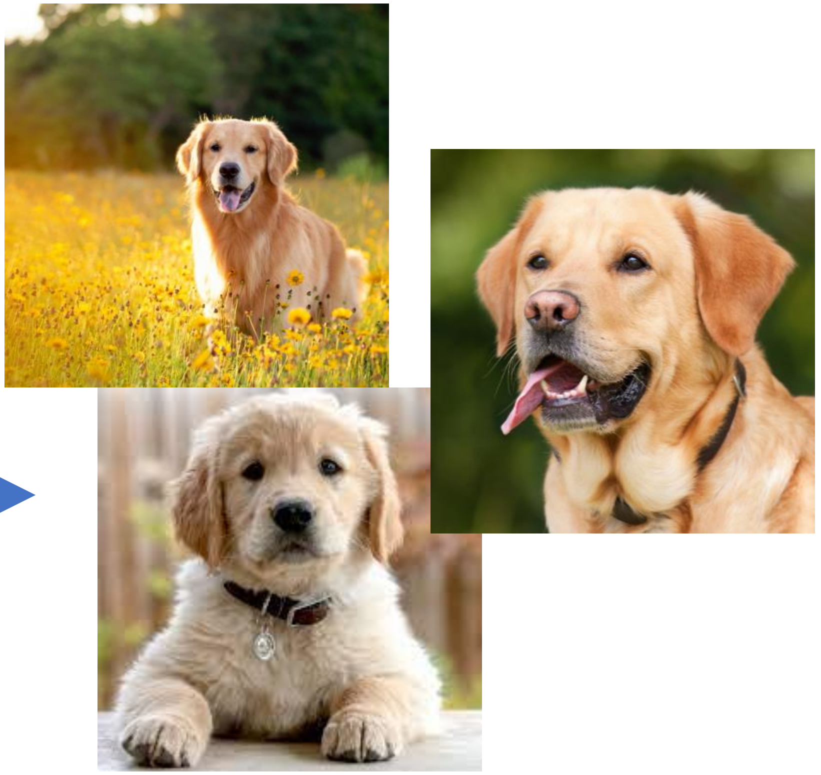
Tanishq Abraham

9/17/2022

A recap

What’s the task?

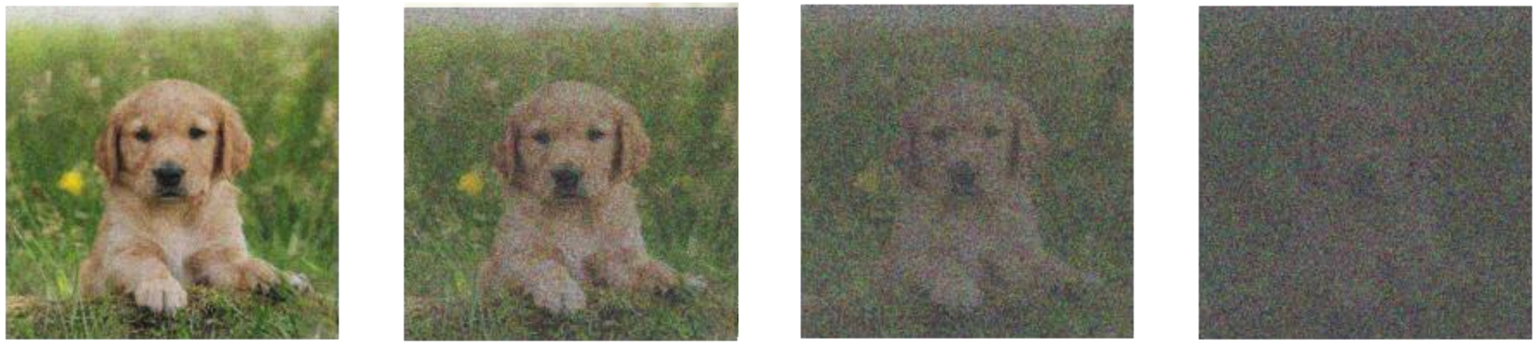

Given these datapoints…


Can we generate more like it?

Forward process

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

$$q(\mathbf{x}_t|\mathbf{x}_0) := \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

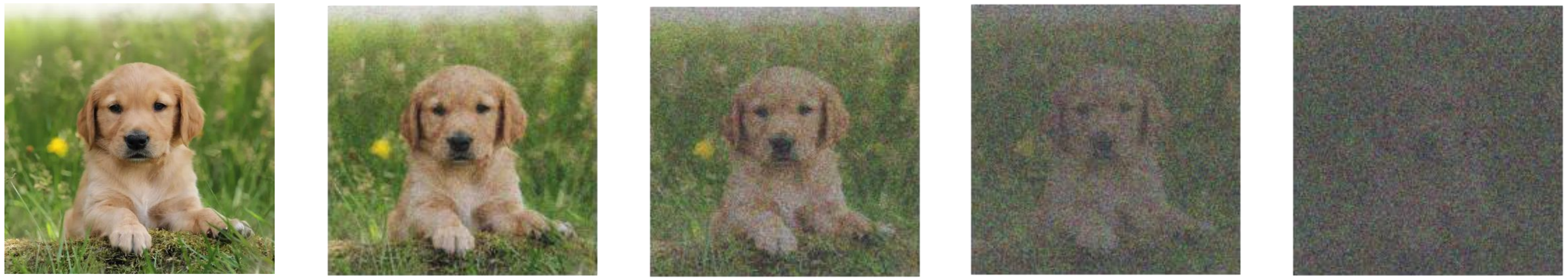

$\mathbf{x}_{25}$
$\mathbf{x}_{0}$ / $\mathbf{x}_{50}$
$\mathbf{x}_{75}$
$\mathbf{x}_{100} = \mathbf{x}_T$

Observed image

Equivalent to Gaussian noise

Reverse process

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) \coloneqq \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \beta_t \mathbf{I})$$



$\mathbf{x}_0$ $\mathbf{x}_{25}$ $\mathbf{x}_{50}$ $\mathbf{x}_{75}$ $\mathbf{x}_{100} = \mathbf{x}_T$

Generated image!

Equivalent to Gaussian noise

Model objective

$$L(\theta) = E_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}[||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)||^2]$$

where:

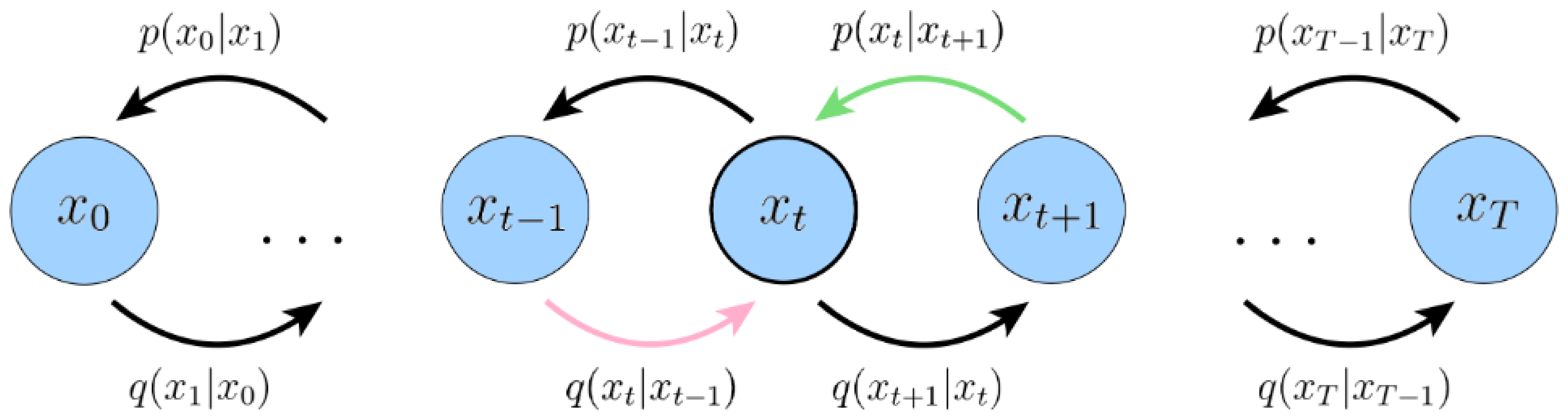
$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Denoising is equivalent to score matching:

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0) = -\frac{1}{(1-\bar{\alpha}_t)}(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}$$

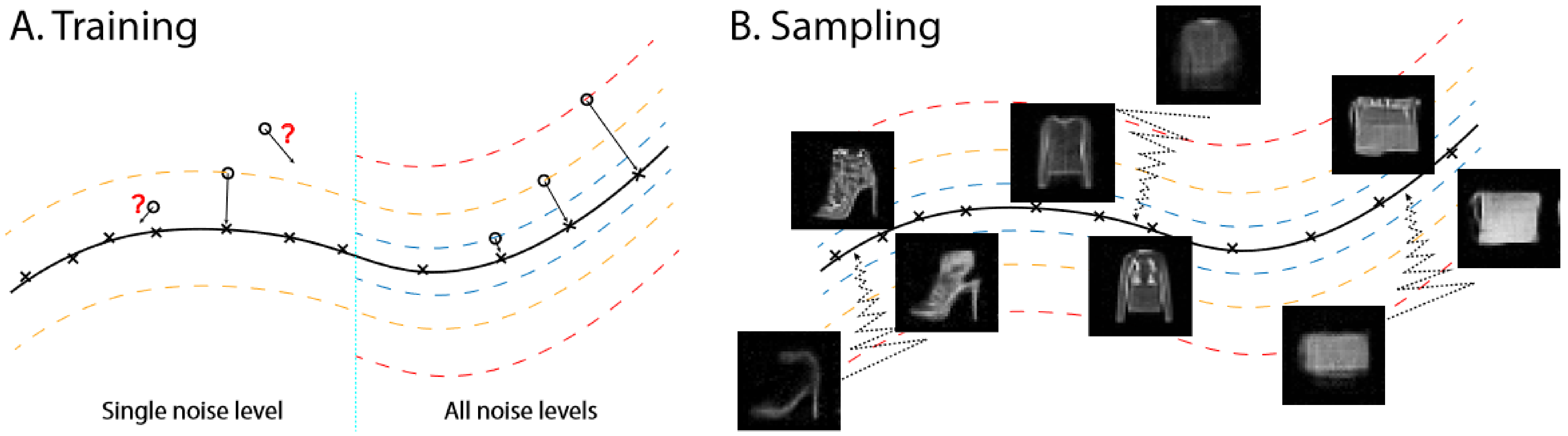
$$\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = -\sqrt{1-\bar{\alpha}_t}\,\mathbf{s}_\theta(\mathbf{x}_t, t)$$

Latent variable perspective – Hierarchical variational autoencoder



A diffusion model can be considered as a hierarchical VAE with a fixed Gaussian encoder

Score matching perspective - Learning the data manifold!



Comparing NCSN and DDPM

DDPM noisy distribution:

$$q(\mathbf{x}_t|\mathbf{x}_0) \coloneqq \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$$

NCSN noisy distribution:

$$q(\mathbf{x}_t|\mathbf{x}_0) \coloneqq \mathcal{N}(\mathbf{x}_t;\ \mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

Sampling differences:

- DDPM decreases the noise scale at each Langevin step (ancestral sampling)
- NCSN runs multiple Langevin steps at each noise scale (annealed Langevin sampling)

Minor architectural differences

Any questions?

Improved techniques for training score-based generative models

It’s time for version 2!

Refamiliarize ourselves with the NCSN notation

Let there be L increasing standard deviations $\sigma_1 < \sigma_2 < \cdots \sigma_i < \cdots < \sigma_L$

Then we have a noisy distribution at each scale:

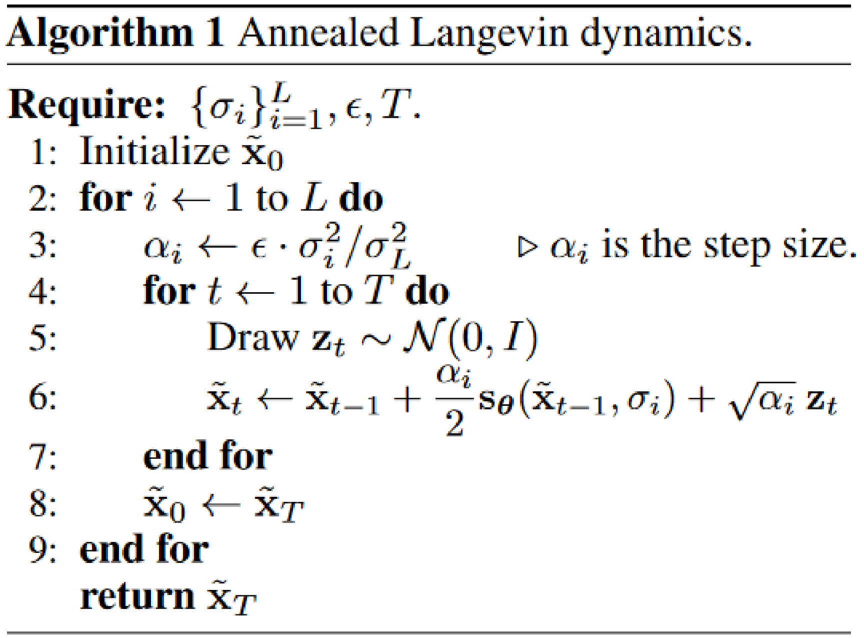
$$p_{\sigma_i}(\tilde{\mathbf{x}}) = \int \mathcal{N}(\tilde{\mathbf{x}}|\mathbf{x}, \sigma_t^2\mathbf{I})p(\mathbf{x})\mathrm{d}\mathbf{x}$$

We train a *single* score network conditioned on the noise scale such that $\mathbf{s}_\theta(\tilde{\mathbf{x}}, \sigma) \approx \nabla_{\mathbf{x}} \log p_{\sigma_i}(\tilde{\mathbf{x}})$, done with the following objective:

$$\mathcal{L}_{ncsn} = \sum_{i=1}^{L} \lambda(\sigma_i)\ell(\theta; \sigma_i)$$

$$\ell(\theta; \sigma_i) = \frac{1}{2} E_{q_{\sigma_i}(\tilde{\mathbf{x}}, \mathbf{x})}\left[\left\|\frac{1}{\sigma^2}(\tilde{\mathbf{x}} - \mathbf{x}) + \mathbf{s}_\theta(\tilde{\mathbf{x}}, \sigma)\right\|_2^2\right]$$

$$\lambda(\sigma_i) = \sigma_i^2$$



Algorithm 1 Annealed Langevin dynamics.

```
Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T$.
1: Initialize $\tilde{\mathbf{x}}_0$
2: for $i \leftarrow 1$ to $L$ do
3:     $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$        ▷ $\alpha_i$ is the step size.
4:     for $t \leftarrow 1$ to $T$ do
5:         Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$
6:         $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$
7:     end for
8:     $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$
9: end for
   return $\tilde{\mathbf{x}}_T$
```

Technique 1

Choose σ_1 to be as large as the maximum Euclidean distance between all pairs of training data points

Intuition: noise should cover the "span" of the entire dataset

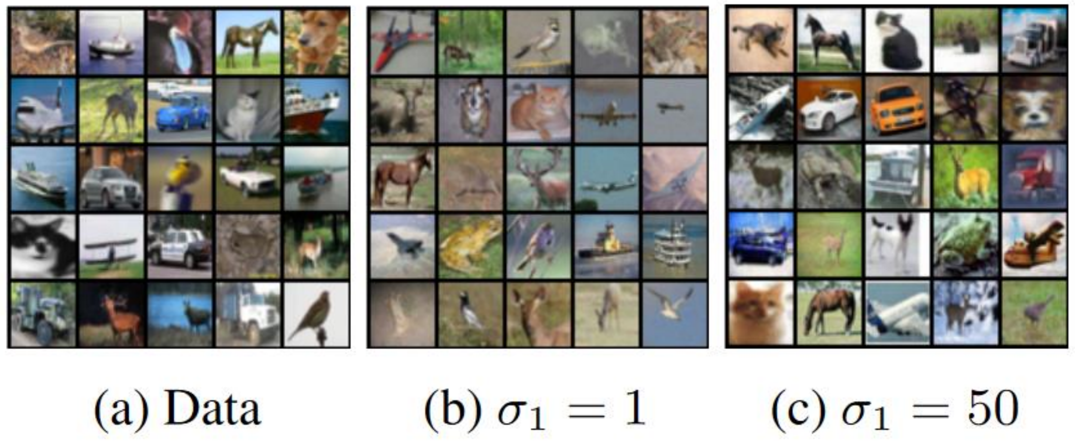


Figure 2: Running annealed Langevin dynamics to sample from a mixture of Gaussian centered at images in the CIFAR-10 test set.

Technique 2

Choose $\{\sigma_i\}_{i=1}^{L}$ as a geometric progression with common ratio γ, such that $\Phi\left(\sqrt{2D}(\gamma-1)+3\gamma\right)-\Phi\left(\sqrt{2D}(\gamma-1)-3\gamma\right)\approx 0.5$

Intuition: You start from noise scale σ_{i-1} and go to σ_i so you want the high-density regions of these distributions to overlap as much as possible for Langevin dynamics to go smoothly

Technique 3

Parametrize the NCSN with $\mathbf{s}_\theta(\mathbf{x}, \sigma) = \frac{\mathbf{s}_\theta(\mathbf{x})}{\sigma}$

Intuition: $\|\mathbf{s}_\theta(\mathbf{x}, \sigma)\|_2 \propto \frac{1}{\sigma}$

Technique 4

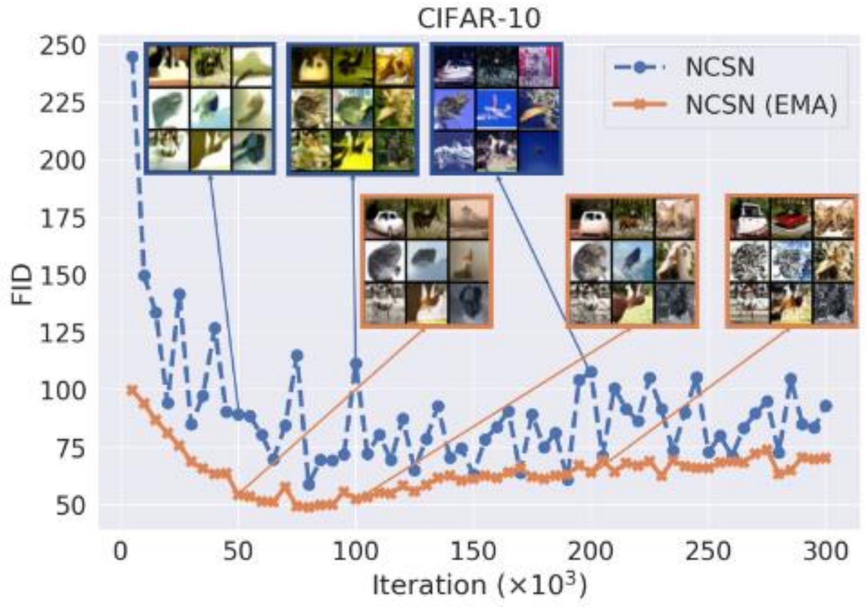
Choose T as large as possible, and then select ϵ that makes $\frac{s_T^2}{\sigma_T^2} \approx 1$

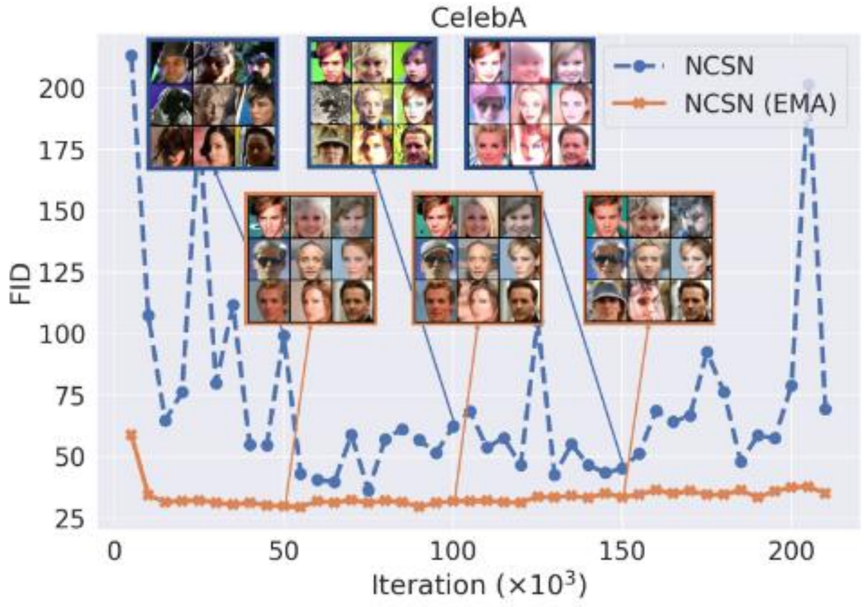
Intuition: $\mathbf{x}_T$ should be a sample from a normal distribution with the correct noise scale

Technique 5

Apply exponential moving average during training

Intuition: While loss is stably decreasing, visual sample quality is unstable





Other experimental details

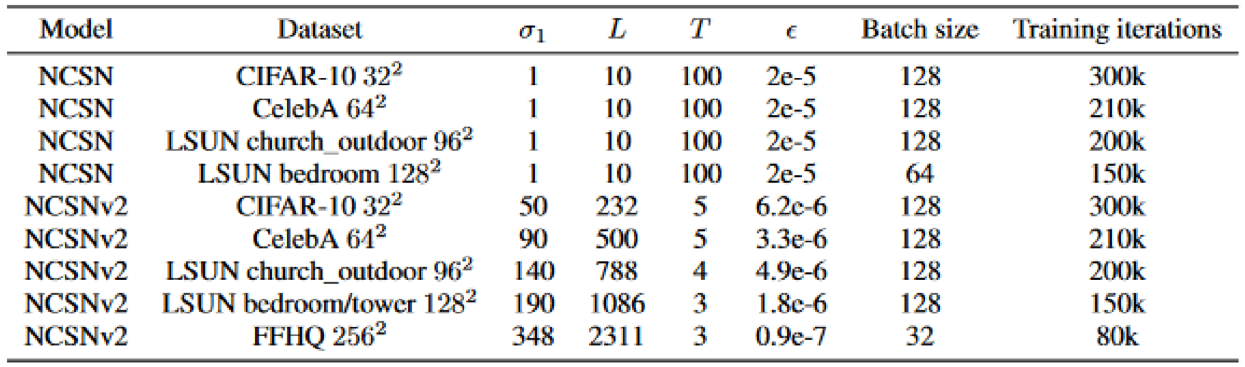
Additional denoising step can be used: $\mathbf{x}_T + \sigma_T^2 \mathbf{s}_\theta(\mathbf{x}_T, \sigma_T)$

RefineNet architecture modified for unconditional score prediction (update instance norm layers to no condition, remove some norm layers)

Adam optimizer with defaults

Lower learning rate with Technique 3

Table 4: Hyperparameters of NCSN/NCSNv2. The latter is configured according to Technique 1–4. σ_1 and L determine the set of noise levels. T and ϵ are parameters of annealed Langevin dynamics.



Model	Dataset	σ_1	L	T	ϵ	Batch size	Training iterations
NCSN	CIFAR-10 32^2	1	10	100	2e-5	128	300k
NCSN	CelebA 64^2	1	10	100	2e-5	128	210k
NCSN	LSUN church_outdoor 96^2	1	10	100	2e-5	128	200k
NCSN	LSUN bedroom 128^2	1	10	100	2e-5	64	150k
NCSNv2	CIFAR-10 32^2	50	232	5	6.2e-6	128	300k
NCSNv2	CelebA 64^2	90	500	5	3.3e-6	128	210k
NCSNv2	LSUN church_outdoor 96^2	140	788	4	4.9e-6	128	200k
NCSNv2	LSUN bedroom/tower 128^2	190	1086	3	1.8e-6	128	150k
NCSNv2	FFHQ 256^2	348	2311	3	0.9e-7	32	80k

Results

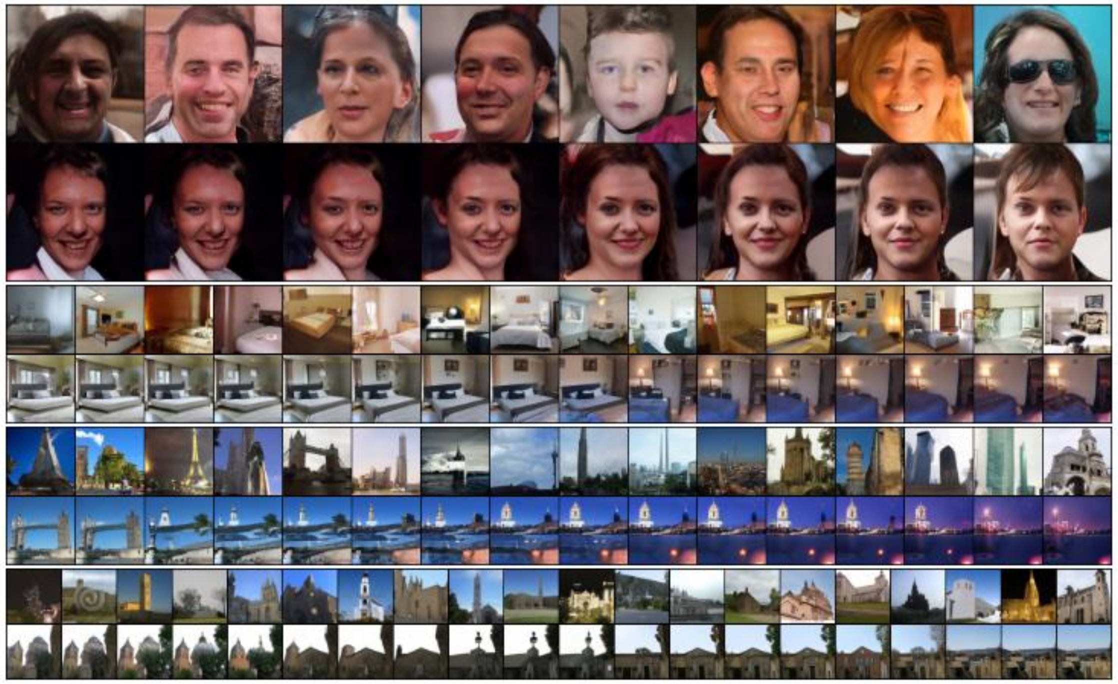


Figure 6: From top to bottom: FFHQ 256^2, LSUN bedroom 128^2, LSUN tower 128^2, and LSUN church_outdoor 96^2. Within each group of images: the first row shows uncurated samples from NCSNv2, and the second shows the interpolation results between the leftmost and rightmost samples with NCSNv2. You may zoom in to view more details.

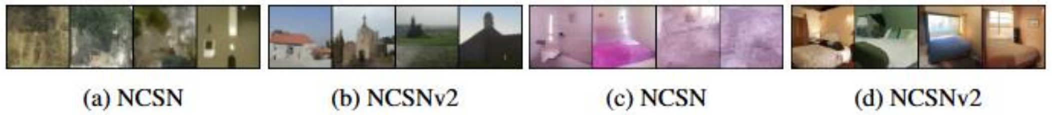


Figure 7: NCSN vs. NCSNv2 samples on LSUN church_outdoor (a)(b) and LSUN bedroom (c)(d).

Table 1: Inception and FID scores.

Model	Inception ↑	FID ↓
CIFAR-10 Unconditional		
PixelCNN [17]	4.60	65.93
IGEBM [18]	6.02	40.58
WGAN-GP [19]	$7.86 \pm .07$	36.4
SNGAN [20]	$8.22 \pm .05$	21.7
NCSN [1]	$\mathbf{8.87 \pm .12}$	25.32
NCSN (w/ denoising)	$7.32 \pm .12$	29.8
NCSNv2 (w/o denoising)	$8.73 \pm .13$	31.75
NCSNv2 (w/ denoising)	$8.40 \pm .07$	**10.87**
CelebA 64 × 64		
NCSN (w/o denoising)	-	26.89
NCSN (w/ denoising)	-	25.30
NCSNv2 (w/o denoising)	-	28.86
NCSNv2 (w/ denoising)	-	**10.23**

What’s next?

Yang Song et al.: Okay that’s great, now what?

Yang Song et al.: Time for a unified formulation!

Yang Song et al.: We have discrete timesteps/noisescales, let’s try one of the oldest tricks in the (mathematician’s) book…

…take the limit and and make continuous!

Score-Based Generative Modeling through Stochastic Differential Equations

Everything’s an SDE!

Ordinary Differential Equations (ODEs)

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) \;\rightarrow \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t$$

Solution:

$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t \mathbf{f}(\mathbf{x}, \tau)\mathrm{d}\tau$$

Numerical solution:

$$\mathbf{x}(t + \Delta t) \approx \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t)\Delta t$$

Stochastic Differential Equations (SDEs)

Like ODEs… *but with noise!*

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \underbrace{\mathbf{f}(\mathbf{x}, t)}_{\text{drift coefficient}} + \underbrace{g(t)}_{\text{diffusion coefficient}} \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} \rightarrow \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

Wiener process (Brownian motion)

Numerical solution:

$$\mathbf{x}(t + \Delta t) \approx \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t)\Delta t + g(t)\mathbf{z}_t$$

where $\mathbf{z}_t \sim \mathcal{N}(0, I)$

SDE formulation of score-based generative modeling

In the limit of $\Delta t \to 0$, our diffusion process can now be described by $\mathbf{x}(t)$, indexed by a continuous time variable $t \in [0, T]$.

$\mathbf{x}(0) \sim p_0(x)$ which is the data distribution

$\mathbf{x}(T) \sim p_T(x)$ which is the prior distribution

Let's denote the probability density of $\mathbf{x}(t)$ as $p_t(\mathbf{x})$ and the transition kernel as $p_{st}(\mathbf{x}(t)|\mathbf{x}(s))$

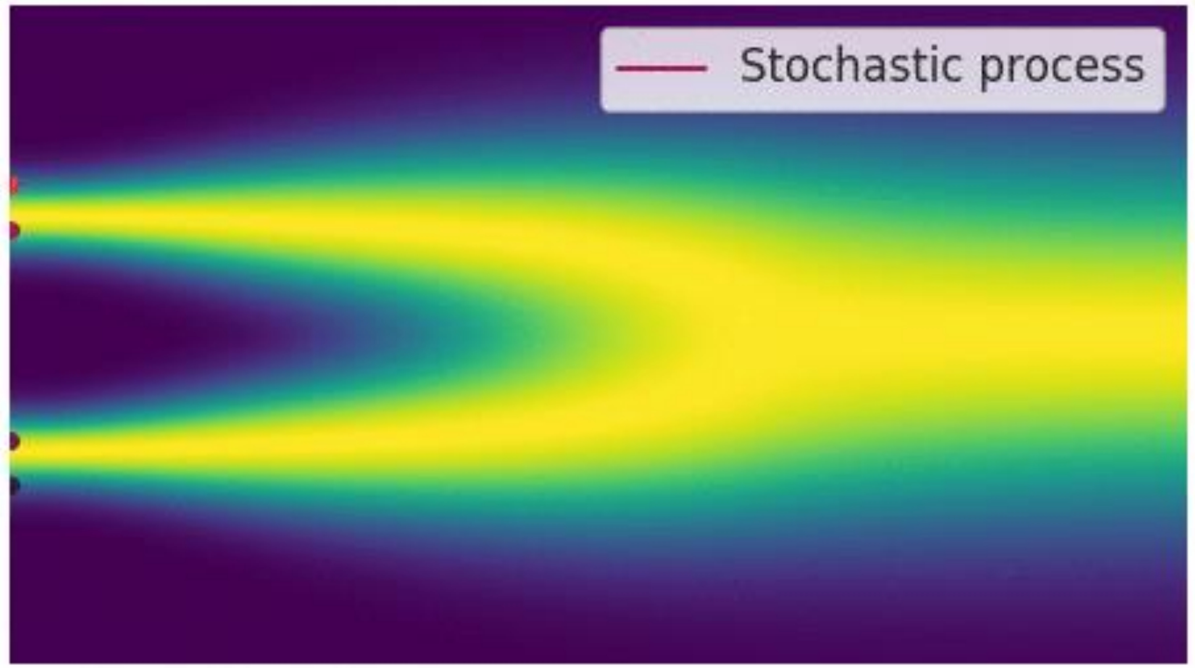
An important property

The reverse of a diffusion process is also a diffusion process, described by what is known as the Anderson reverse-time SDE:

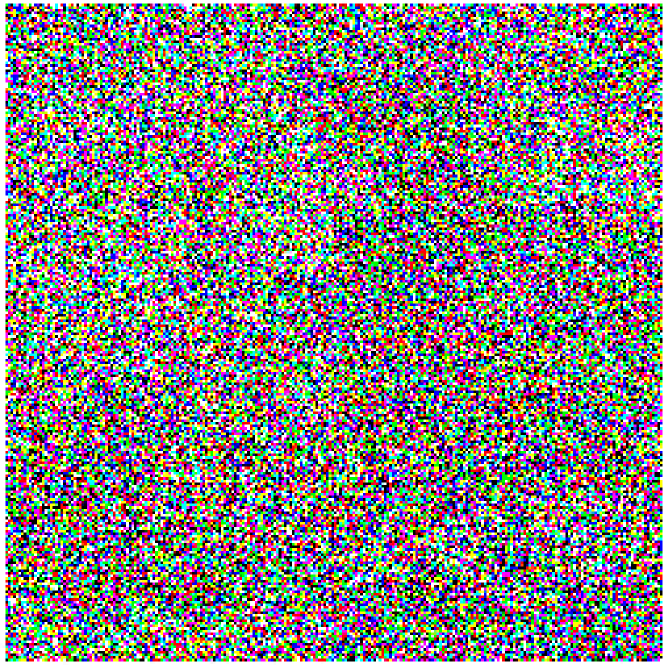
$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$$

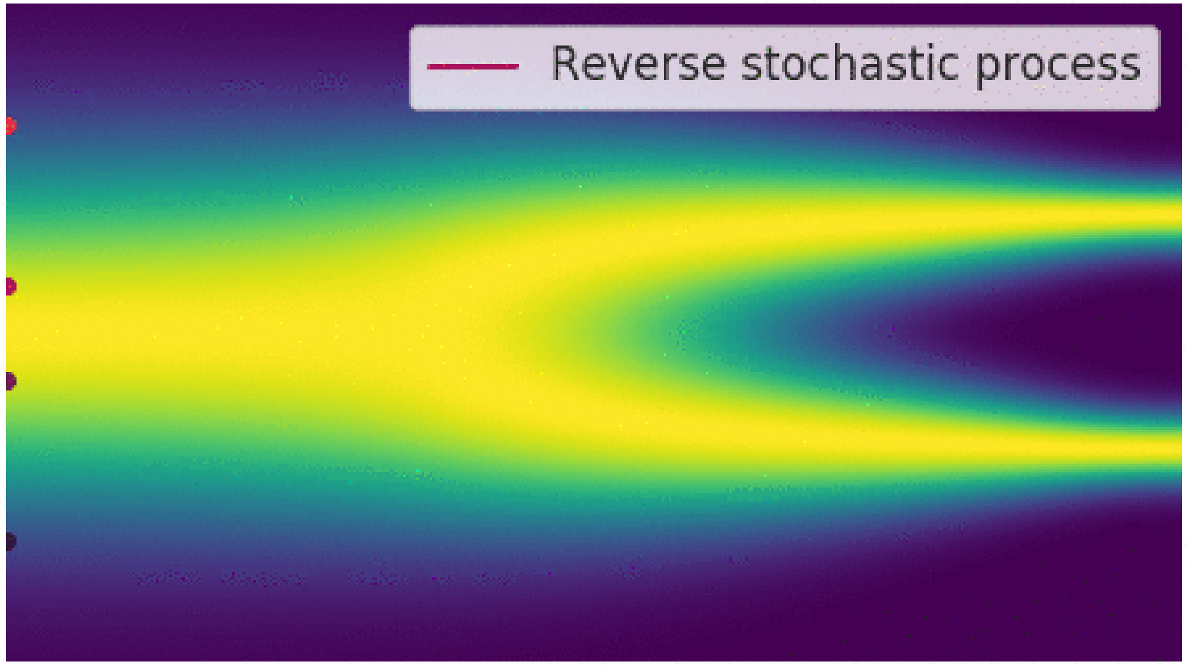
where $\bar{\mathbf{w}}$ is the Wiener process for time flowing backward from T to 0

Forward process as modeled by an SDE



Reverse process as modeled by a reverse-time SDE





NCSN – a variance-exploding (VE) SDE

We can describe NCSN forward process with the following Markov chain:

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2}\,\mathbf{z}_{i-1}, \qquad i = 1, \cdots, N$$

If we define continuous functions such that $\mathbf{x}\left(\frac{i}{N}\right) = \mathbf{x}_i$, $\sigma\left(\frac{i}{N}\right) = \sigma_i$, and $\mathbf{z}\left(\frac{i}{N}\right) = \boldsymbol{z}_i$ then we can rewrite the equation as:

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \sqrt{\sigma^2(t + \Delta t) - \sigma^2(t)}\,\mathbf{z}(t) \approx \mathbf{x}(t) + \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}\Delta \mathrm{t}}\,\mathbf{z}(t)$$

In the limit of $\Delta t \to 0$, we get the following SDE:

$$\mathrm{d}\mathbf{x} = \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}}\,\mathrm{d}\mathbf{w}$$

DDPM – a variance-preserving (VP) SDE

We can describe DDPM forward process with the following Markov chain:

$$\mathbf{x}_i = \sqrt{1-\beta_i}\mathbf{x}_{i-1} + \sqrt{\beta_i}\,\mathbf{z}_{i-1}, \qquad i = 1, \cdots, N$$

Through a similar math derivation, the corresponding SDE is:

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}$$

Sub-variance preserving (Sub-VP) SDE

Authors also proposed a new SDE:

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)\left(1 - e^{-2\int_0^t \beta(s)\mathrm{d}s}\right)}\mathrm{d}\mathbf{w}$$

Variance is bounded by the corresponding VP SDE

Training the score function for the reverse SDE

The training objective with denoising score matching is as follows:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_t \Big\{ \lambda(t) \mathbb{E}_{\mathbf{x}(0)} \mathbb{E}_{\mathbf{x}(t)\mid\mathbf{x}(0)} \Big[\big\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0)) \big\|_2^2 \Big] \Big\}.$$

$$p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0)) = \begin{cases} \mathcal{N}\big(\mathbf{x}(t); \mathbf{x}(0), [\sigma^2(t) - \sigma^2(0)]\mathbf{I}\big), & \text{(VE SDE)} \\ \mathcal{N}\big(\mathbf{x}(t); \mathbf{x}(0)e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, \mathbf{I} - \mathbf{I}e^{-\int_0^t \beta(s)\mathrm{d}s}\big) & \text{(VP SDE)} \\ \mathcal{N}\big(\mathbf{x}(t); \mathbf{x}(0)e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, [1 - e^{-\int_0^t \beta(s)\mathrm{d}s}]^2\mathbf{I}\big) & \text{(sub-VP SDE)} \end{cases}.$$