Next Week Guest Lectures on Zoom

Score-Based Models and Diffusion Models

Recap: Boltzmann Machine Training

- Objective: maximum likelihood learning (assume T =1):
  - Probability of one sample:

$y \in \mathbb{R}^d$

$$P(y) = \frac{\exp(\frac{1}{2}y^\top W y)}{\underbrace{\sum_{y'} \exp(y'^\top W y')}_{Z_\theta}}$$

  - Maximum log-likelihood:

$$L(W) = \frac{1}{N}\sum_{y \in D} \frac{1}{2} y^\top W y - \log \sum_{y'} \exp(\frac{1}{2} y'^\top W y')$$

Can we avoid calculating the gradient of normalizing constant ($\nabla_x Z_\theta$)?

Z_θ : normalizing constant

Score Matching

model $p(x)$
↓
model $\nabla_x \log p_{data}(x)$

- Score Function
  - Definition:

$$\nabla_x \log p_{data}(x) : \mathbb{R}^d \rightarrow \mathbb{R}^d$$

- Idea: directly fitting the score function:
  - $\min_{\theta} \mathbb{E}_{p_{data}} \|\nabla_x \log p_\theta(x) - \nabla_x \log p_{data}(x)\|^2$
  - No need to compute $\nabla_x Z_\theta$!

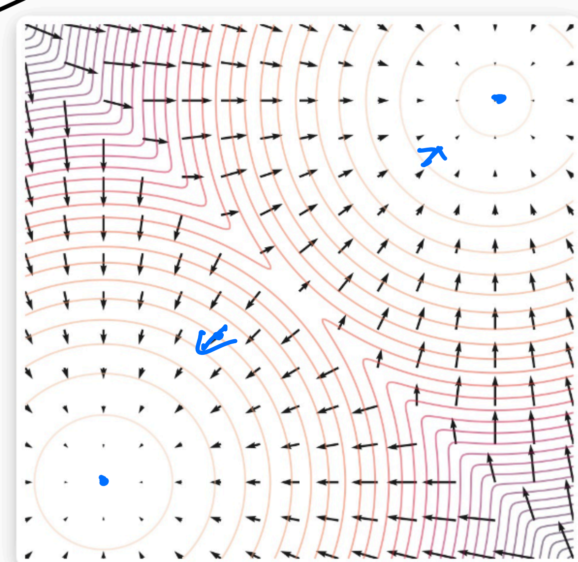
$$p(x) = \frac{e^{-f_\theta(x)}}{\sum_{x'} e^{-f_\theta(x')}}$$

$$\log p(x) = -f_\theta(x) - \sum_{x'} f_\theta(x')$$

$$\nabla \log p(x) = -\nabla f_\theta(x)$$

- Problem:
  - How to compute $\nabla_x \log p_{data}(x)$?

$\{x_1, \dots, x_N\}$



Score function (the vector field) and density function (contours) of a mixture of two Gaussians.

Score Matching

$$\mathbb{E}_{p_{data}} \| \nabla_x \log p_\theta(x) - \nabla_x \log p_{data}(x) \|^2$$

$$= \underline{\mathbb{E}_{p_{data}} \| \nabla_x \log p_\theta(x) \|^2} + \mathbb{E}_{p_{data}} \| \nabla_x \log p_{data}(x) \|^2$$

$$- 2\, \mathbb{E}_{p_{data}} \langle \nabla_x \log p_\theta(x), \nabla_x \log p_{data}(x) \rangle$$

(Integration by parts

$$\mathbb{E}_p \langle f(x), \nabla_x \log p(x) \rangle = - \mathbb{E}_p [\operatorname{div} f(x)]$$

where $\operatorname{div} f(x) = \sum_i \frac{\partial f_i(x)}{\partial x_i}$)

$$\Rightarrow \mathbb{E}_{p_{data}} \langle \nabla_x \log p_\theta(x), \nabla_x \log p_{data}(x) \rangle = - \mathbb{E} [\mathrm{Tr} (\nabla_x^2 \log p_\theta(x))]$$

$$\Rightarrow \text{loss} \Leftrightarrow \mathbb{E}_{p_{data}} \| \nabla_x \log p_\theta(x) \|_2^2 - 2\, \mathbb{E}_{p_{data}} [\mathrm{Tr} (\nabla_x^2 \log p_\theta(x))]$$

Score Matching

use NN to parameterize $\nabla_x \log p_\theta(x)$: $s_\theta(x): \mathbb{R}^d \to \mathbb{R}^d$

loss: $$\frac{1}{N} \sum_{\text{training } x_i} \underbrace{\| s_\theta(x_i) \|_2^2}_{O(d)} - 2\left[\underbrace{\mathrm{Tr}(\nabla s_\theta(x_i))}_{O(d^3)} \right]$$

Sliced Score Matching

$A\sigma(W\sigma(Wx))$

$$L(\theta) = \frac{1}{N}\sum_{x\in D} \|s_\theta(x)\|^2 - 2\left[Tr(Ds_\theta(x))\right]$$

random projection

let $M \in \mathbb{R}^{d\times d}$, v random, $\mathbb{E}[vv^T] = I$, $\mathbb{E}_v[v^T M v] = \mathbb{E}[Tr(Mvv^T)]$

$= Tr(M)$

Sample $v_1, \dots, v_K$

$v_1^T M v_1, \dots v_K^T M v_K$

$K << d$

Score Matching: Langevin Dynamics

x_0 $\quad\quad\quad$ $\epsilon << 1$

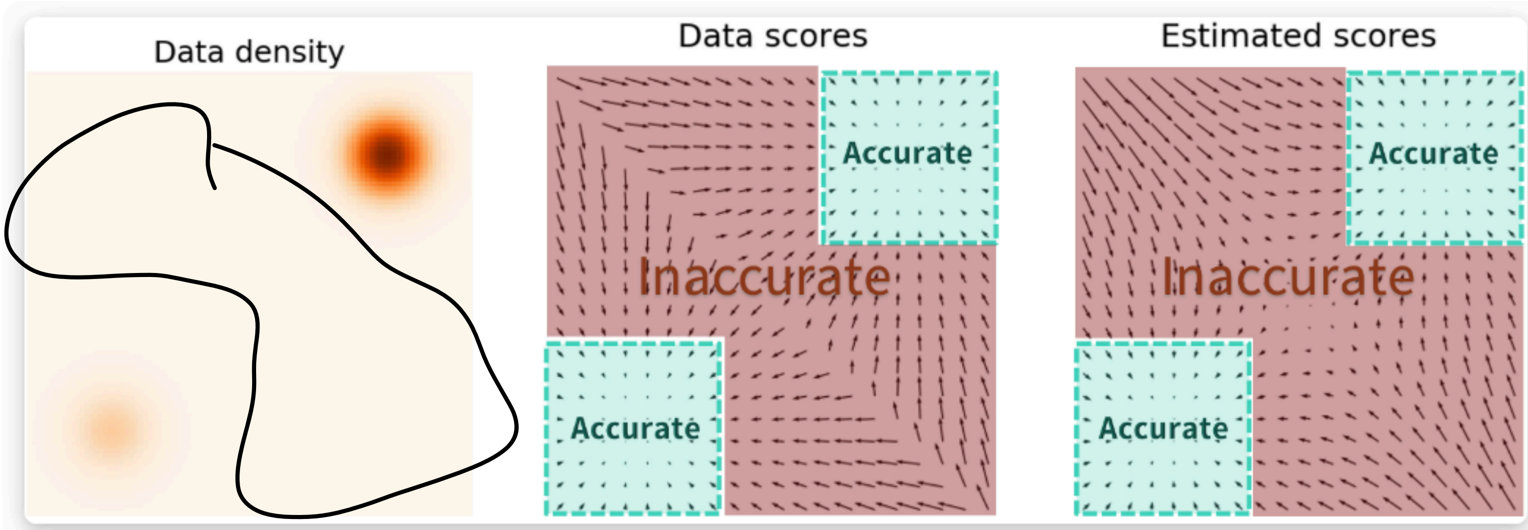
$$x_{t+1} \leftarrow x_t + \epsilon \nabla_x \log p(x) + \sqrt{2\epsilon} z_t, z_t \sim N(0, I)$$

Stationary (equilibrium distribution): p(x)

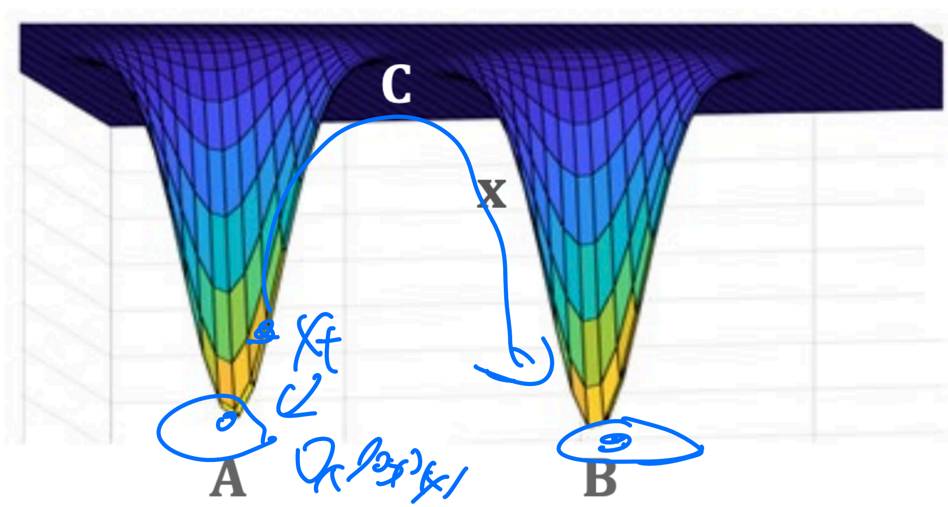
$\{x_1, \ldots, x_\infty\} \rightarrow p(x)$

Practical Issues

- Score function estimation is inaccurate in low density regions (few data available).



- Sampling is Slow



Annealing: Denoising Score Matching

- Fit several “smoothed” versions of p_{data}:
  - Choose temperatures: $\sigma_1, \sigma_2, \ldots, \sigma_T$
  - $p_{\sigma_i,data}(x) = \underline{p_{data}(x) * N(0,\sigma_i)} = \int_\delta p_{data}(x-\delta)N(x;\delta,\sigma_i)d\delta$

    convolution
  - Implementation:
    - Take a sample x, draw a sample $z \sim N(0,\sigma_i)$, output $x' = x + z$.

$$\sigma_1 > \sigma_2 > \cdots > \sigma_{L-1} > \sigma_L$$

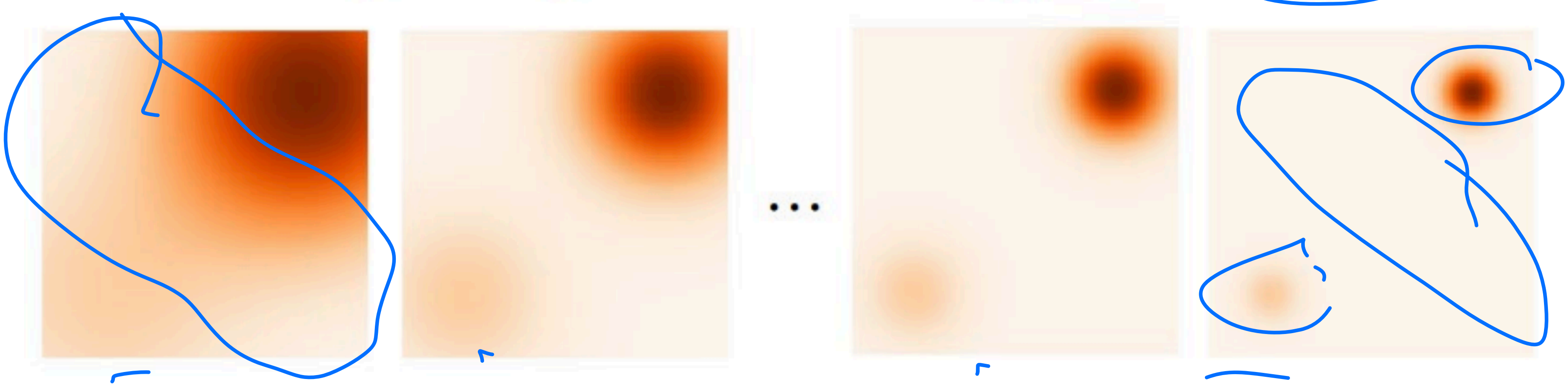


Figure by Stefano Ermon.

Annealing: Denoising Score Matching

$$\arg\min_{\theta} \sum_{i} \lambda(\sigma_i)\mathbb{E}_{x\sim p_{\sigma_i,data}}\|s_\theta(x,i) - \nabla_x \log p_{\sigma_i,data}(x)\|^2$$

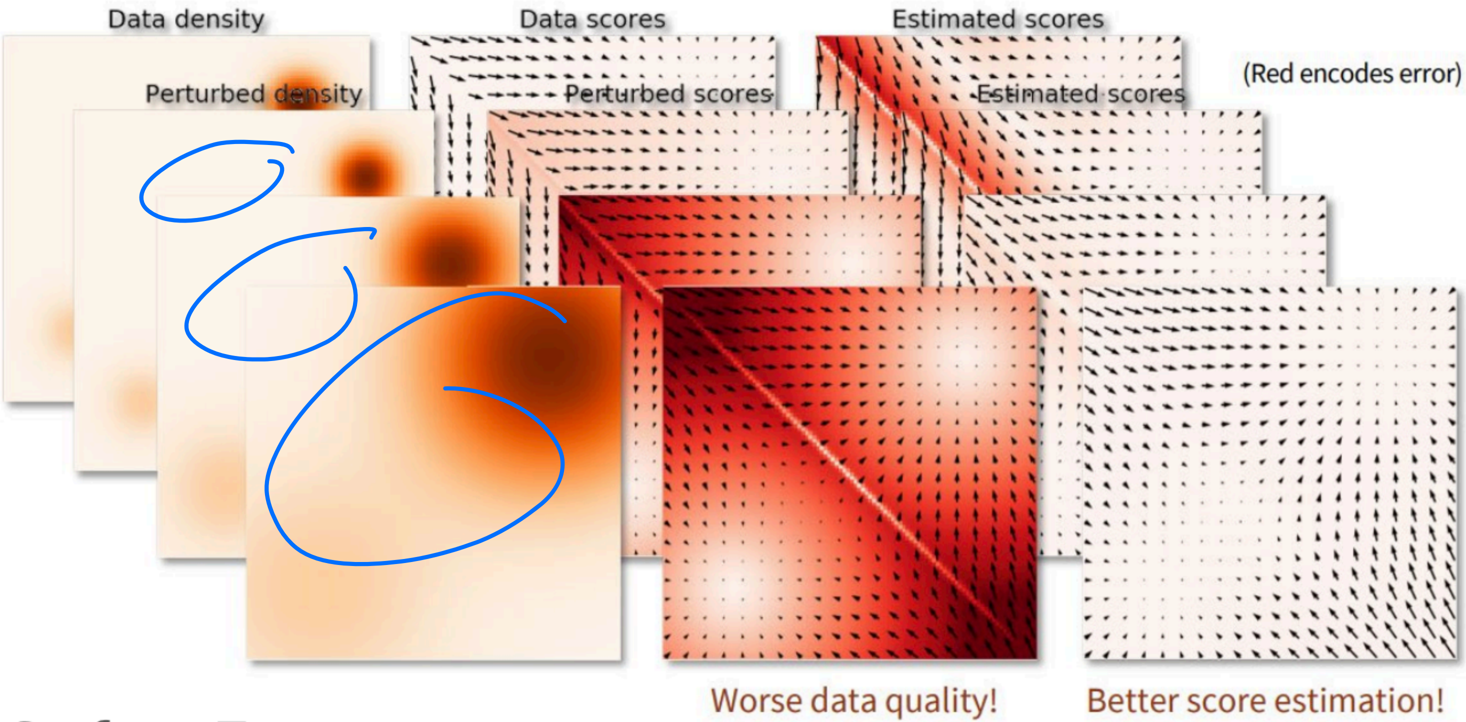


Figure by Stefano Ermon.

Annealed Langevin Dynamics

Algorithm 1 Annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T.$

Initialize $\tilde{\mathbf{x}}_0$

for $i \leftarrow 1$ to L **do**

  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2 / \sigma_L^2$ ▷ α_i is the step size.

  for $t \leftarrow 1$ to T **do**

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

    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2} \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\, \mathbf{z}_t$

  end for

  $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$

end for

return $\tilde{\mathbf{x}}_T$

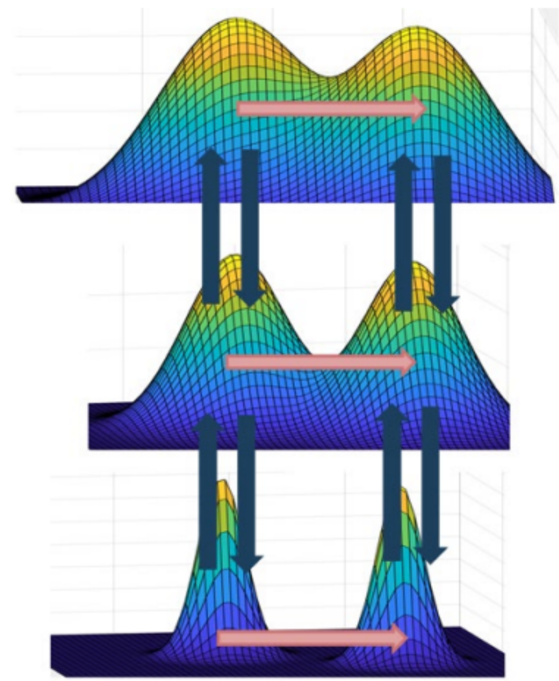


Figure from Song-Ermon '19

Diffusion Models

An image generated by Stable Diffusion based on the text prompt "a photograph of an astronaut riding a horse"

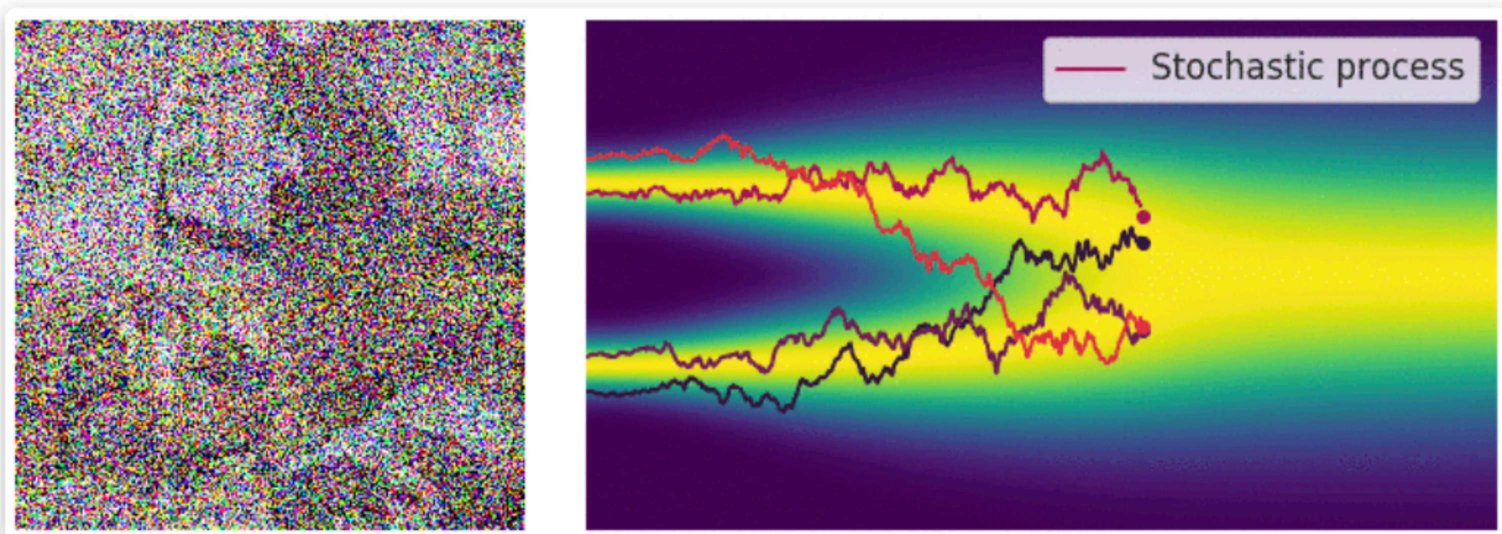
Perturbing Data with an SDE

$\sigma_1 < \sigma_2 \cdots \sigma_T$

- Let the number of noise scales approaches infinity!

$\sigma_1, \ldots, \sigma_T$ T large, $X_T = x + \mathcal{N}(0, \sigma_T)$

$T \to \infty$, $\{\sigma_t\}_t$,

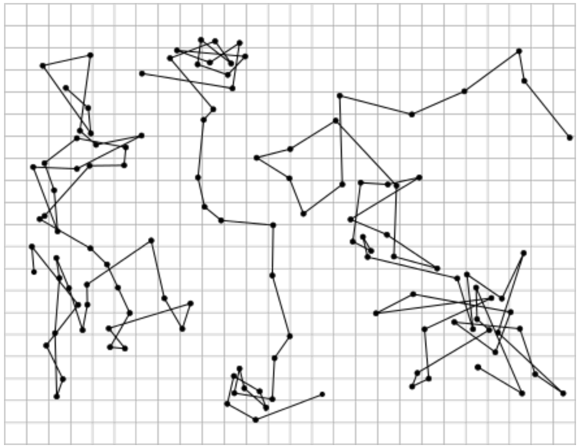


Perturbing data to noise with a continuous-time stochastic process.

Stochastic Differential Equations

$$dx = \underbrace{f(x,t)dt}_{} + g(t)dw$$

- x(0): real image, x(T): Gaussian noise.
- f(x,t): drift terms. g(t): diffusion coefficient.
- dw: Brownian motion
  - $w(t+u) - w(t) \sim N(0,u)$



- f(x,t) and g(t) are parts of the model.
  - Variance Exploding SDE: $dx = \sqrt{\frac{d[\sigma^2(t)]}{dt}}dw.$
  - Variance Preserving SDE: $dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)}dw.$
  - $\sigma(t), \beta(t)$ are hyper-parameters.

exp(t)

Reversing the SDE

- Reversing the SDE: finding some stochastic process that goes from noise to data.
  - Use to generate data!

- Theorem (Anderson '82): there exists a reversing SDE, and it has a nice form:

$$dx = [f(x,t) - g^2(t)\,\nabla_x \log p_t(x)]dt + g(t)dw$$

- Strategy: learn the score function, then solve this reverse SDE.

$$\begin{array}{c} X(T) \sim \mathcal{N}(0, I) \\ \downarrow \\ X(0) \end{array}$$

Reversing the SDE

- Learning the score function: use score matching!

$$\arg\min_{\theta} \sum_{i} \lambda(\sigma_i) \mathbb{E}_{x \sim p_{\sigma_i,data}} \|s_\theta(x, i) - \nabla_x \log p_{\sigma_i,data}(x)\|^2$$

$$\Rightarrow \arg\min_{\theta} \mathbb{E}_{t \sim unif[0,T]} \mathbb{E}_{p_t(x)} \left[\lambda(t) \|s_\theta(x, t) - \nabla_x \log p_t(x)\|^2\right]$$

- Use existing techniques: sliced score matching

- No need to tune temperature schedule
  - Still need to choose a forward SDE, $\lambda(\sigma_i)$, etc
  - Typically choose $\lambda(t) \propto 1/\mathbb{E}\left[\|\lambda_{x(t)} \log p(x(t) \mid x(0))\|^2\right]$

Sampling by Solving the Reverse SDE

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

$$dx = [f(x,t) - g^2(t)\,\nabla_x \log p_t(x)]dt + g(t)dw$$

- Euler-Maruyama discretization:
  - $\Delta x \leftarrow [f(x,t) - g^2(t)s_\theta(x,t)]\Delta t + g(t)\sqrt{\Delta t}z_t$
  - $x \leftarrow x + \Delta x$
  - $t \leftarrow t + \Delta t$

- Other solvers:
  - Runge-Kutta
  - Predictor-corrector (Song et al. '21)

Evaluating Probability by Converting to ODE

- De-randomizing SDE

$$dx = [f(x,t) - g^2(t)\nabla_x \log p_t(x)]dt + g(t)dw$$

$X(T)$

$$\downarrow$$

$$dx = [f(x,t) - g^2(t)\nabla_x \log p_t(x)]dt, x(T) \sim p_T$$

- Given an initial distribution and an ODE, we can evaluate probability at any time
    - Say given $x(T) \sim p_T$ and $dx = f(x,t)dt$

$$\log p_0(x(0)) = \log p_T(X(T)) + \int_0^T Tr(Df_\theta(x,t))dt$$

- Solve via ODE.