

Generative Models*

- Inference vs Generation
- Monte Carlo vs Generator Methods
- Gibbs Distributions
- Monte Carlo Markov Chains

*See this helpful blog for an overview: Yang Song, “Generative Modeling by Estimating Gradients of the Data Distribution,” web blog post, May 5, 2021, <https://yang-song.net/blog/2021/score> .

Inference vs Generation

- Two primary goals in deep learning
 - How to generate random vectors with a desired distribution?
 - Can we learn the distribution from sample data

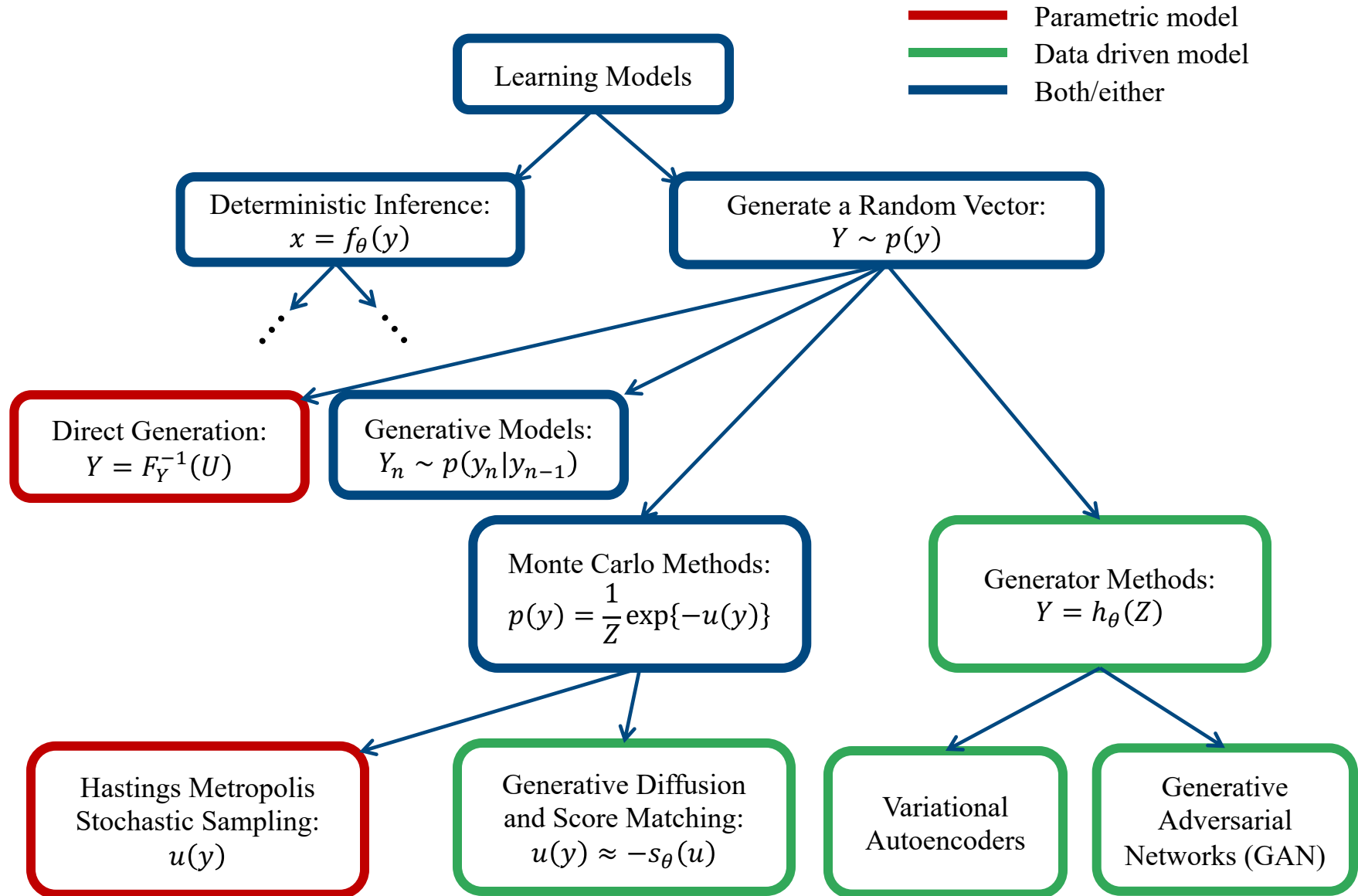
Learn
inference function:
 $x = f_{\theta}(y)$

Goal: Predict
unknown quantity.

Learn
random vector generation:
 $Y \sim p(y)$

Goal: Generate
random vectors.

Taxonomy of Learning Models



Gibbs Distribution

- Let $X \sim p(x)$ be a random object (i.e., image, video, speech).
- Typically, X is assumed to have a Gibbs distribution given by

$$p(x) = \frac{1}{z} \exp\{-u(x)\}$$

- where $u(x)$ is the energy function, and z is the partition function given by $z = E[\exp\{-u(X)\}]$.
- Facts:
 - $u(x) = -\log p(x)$ always exists as long as $p(x) > 0$.
 - z is usually intractable to compute, but that's OK.
 - $u(x)$ increases $\Rightarrow p(x)$ decreases
 - $u(x)$ decreases $\Rightarrow p(x)$ increases
- From Thermodynamics:
 - Also known as Boltzmann distribution
 - The distribution of any system in thermodynamic equilibrium

Monte Carlo Markov Chains

- You can generate a sample from any Gibbs distribution using the Metropolis algorithm given by

Initialize X

Repeat {

 Generate a new proposal $\tilde{X} \sim q(\tilde{x}|X)$

 If $u(\tilde{X}) \leq u(X)$, then $X \leftarrow \tilde{X}$

 else {

$\Delta E \leftarrow u(\tilde{X}) - u(X)$

$p \leftarrow \exp\{-\Delta E\}$

 With probability p , $X \leftarrow \tilde{X}$

 }

}

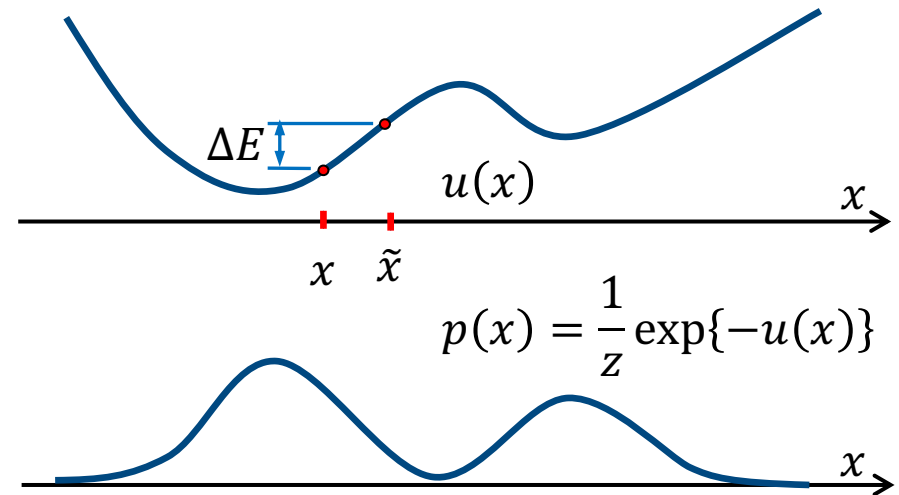
- Notice that:

- Proposal distribution must have the property that $q(\tilde{x}|x) = q(x|\tilde{x})$
- Algorithm only depends on change of $u(x)$
- You don't need to know the partition function, z

Stochastic Sample of Gibbs Distribution

■ Gibbs distribution

- $u(x)$: Energy function
- $p(x)$: Probability density



■ Interpretation

- Proposals that reduce energy are always accepted
- Proposals that increase energy are sometimes accepted.

■ Problem:

- Requires a parametric expression for $u(x)$.


Data Driven Stochastic Sampling?

■ Two approaches to modeling:

- Parametric model (traditional):
 - Human design; small number of parameters; often a physics model
 - Example: $u_{\theta}(x) = \sum_{\{i,j\}} \theta_{i,j} |x_i - x_j|$
- Data Driven model (proposed):

$$\left\{ \begin{array}{c} \{X_0, \dots, X_{K-1}\} \\ \text{training samples} \end{array} \right\} \Rightarrow u_{\theta}(x)$$

deep neural network



■ Great idea, but...

- How do we train a DNN to fit the $u(x)$ that describes training data?
- We don't even know $u(x)$!
- This reduces the problem to an inference problem.
- But what loss function should we use?

■ Solution: Score Matching

Score Matching

- The Score
- Denoising Score Matching
- Geometric Interpretation

Defining the Score[†]

- Let $X \sim p(x)$ be a random object, then we define

- Log probability is given by[†]:

$$l(x) = \log p(x) = -u(x) + c$$

- The score is given by[†]:

$$s(x) = \nabla_x \log p(x) = -\nabla_x u(x)$$

- Important ideas:

- If you know $s(x)$, then you know $u(x)$.
- $s(x)$ is a conservative vector field $\Leftrightarrow [\nabla_x s(x)]^t = \nabla_x s(x)$

[†]Definitions are given assuming a Bayesian estimation framework. The more traditional Frequentist framework uses slightly different definitions and terminology.

Score Matching

- Let $X \sim p(x) = \frac{1}{Z} \exp\{-u(x)\}$:

- Then we can learn the score, $s_\theta(x)$, from data via

$$\hat{\theta} = \arg \min_{\theta} L_{SM}(\theta)$$

- where

$$L_{SM}(\theta) = E \left[\frac{1}{2} \| -\nabla_x u(X) - s_\theta(X) \|^2 \right]$$

- Then we have that:

- $s_{\hat{\theta}}(x)$ is an estimate of the score
- But it may not be a conservative vector field.

- Important Questions:

- Where do we get $\nabla_x u(x)$?
- Can we use $s(x)$ to sample from the Gibbs distribution $p(x)$?

Denoising Score Matching: Theorem*

■ Theorem (Vincent):

- $X \sim p(x) = \frac{1}{Z} \exp\{-u(x)\}$ Gibbs distribution of X
- $\tilde{X}|X \sim q_\sigma(\tilde{x}|x)$ Proposal distribution[†]
- $\tilde{X} \sim p_\sigma(\tilde{x}) = \frac{1}{Z} \exp\{-u_\sigma(x)\}$ Gibbs distribution of \tilde{X}

and define:

- $L_{SM}(\theta; \sigma) = E \left[\frac{1}{2} \left\| -\nabla_{\tilde{x}} u_\sigma(\tilde{X}) - s_\theta(\tilde{X}) \right\|^2 \right]$
- $L_{DSM}(\theta; \sigma) = E \left[\frac{1}{2} \left\| \nabla_{\tilde{x}} \log q_\sigma(\tilde{X}|X) - s_\theta(\tilde{X}) \right\|^2 \right].$

Then

$$L_{SM}(\theta; \sigma) = L_{DSM}(\theta; \sigma) + C$$

Proof: Clever but straight forward. See reference.

*[P. Vincent. A connection between score matching and denoising autoencoders. Neural Computation, 23\(7\):1661–1674, 2011.](#)

[†]We assume the technical conditions that $q_\sigma(\tilde{x}|x)$ is continuously differentiable w.r.t. \tilde{x} and $\forall x, \tilde{x}, q_\sigma(\tilde{x}|x) > 0$.

Proof of Denoising Score Matching Theorem*

Appendix

Proof that $J_{ESMq_\sigma} \sim J_{DSMq_\sigma}$ (11)

The explicit score matching criterion using the Parzen density estimator is defined in Eq. 7 as

$$J_{ESMq_\sigma}(\theta) = \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})} \left[\frac{1}{2} \left\| \psi(\tilde{\mathbf{x}}; \theta) - \frac{\partial \log q_\sigma(\tilde{\mathbf{x}})}{\partial \tilde{\mathbf{x}}} \right\|^2 \right]$$

which we can develop as

$$J_{ESMq_\sigma}(\theta) = \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})} \left[\frac{1}{2} \|\psi(\tilde{\mathbf{x}}; \theta)\|^2 \right] - S(\theta) + C_2 \quad (16)$$

where $C_2 = \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})} \left[\frac{1}{2} \left\| \frac{\partial \log q_\sigma(\tilde{\mathbf{x}})}{\partial \tilde{\mathbf{x}}} \right\|^2 \right]$ is a constant that does not depend on θ , and

$$\begin{aligned} S(\theta) &= \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})} \left[\left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial \log q_\sigma(\tilde{\mathbf{x}})}{\partial \tilde{\mathbf{x}}} \right\rangle \right] \\ &= \int_{\tilde{\mathbf{x}}} q_\sigma(\tilde{\mathbf{x}}) \left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial \log q_\sigma(\tilde{\mathbf{x}})}{\partial \tilde{\mathbf{x}}} \right\rangle d\tilde{\mathbf{x}} \\ &= \int_{\tilde{\mathbf{x}}} q_\sigma(\tilde{\mathbf{x}}) \left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial}{\partial \tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) \right\rangle d\tilde{\mathbf{x}} \\ &= \int_{\tilde{\mathbf{x}}} \left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial}{\partial \tilde{\mathbf{x}}} q_\sigma(\tilde{\mathbf{x}}) \right\rangle d\tilde{\mathbf{x}} \\ &= \int_{\tilde{\mathbf{x}}} \left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial}{\partial \tilde{\mathbf{x}}} \int_{\mathbf{x}} q_0(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) d\mathbf{x} \right\rangle d\tilde{\mathbf{x}} \\ &= \int_{\tilde{\mathbf{x}}} \left\langle \psi(\tilde{\mathbf{x}}; \theta), \int_{\mathbf{x}} q_0(\mathbf{x}) \frac{\partial q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} d\mathbf{x} \right\rangle d\tilde{\mathbf{x}} \\ &= \int_{\tilde{\mathbf{x}}} \left\langle \psi(\tilde{\mathbf{x}}; \theta), \int_{\mathbf{x}} q_0(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} d\mathbf{x} \right\rangle d\tilde{\mathbf{x}} \\ &= \int_{\tilde{\mathbf{x}}} \int_{\mathbf{x}} q_0(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) \left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} \right\rangle d\mathbf{x} d\tilde{\mathbf{x}} \\ &= \int_{\tilde{\mathbf{x}}} \int_{\mathbf{x}} q_\sigma(\tilde{\mathbf{x}}, \mathbf{x}) \left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} \right\rangle d\mathbf{x} d\tilde{\mathbf{x}} \\ &= \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}}, \mathbf{x})} \left[\left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} \right\rangle \right]. \end{aligned}$$

Substituting this expression for $S(\theta)$ in Eq. 16 yields

$$\begin{aligned} J_{ESMq_\sigma}(\theta) &= \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})} \left[\frac{1}{2} \|\psi(\tilde{\mathbf{x}}; \theta)\|^2 \right] \\ &\quad - \mathbb{E}_{q_\sigma(\mathbf{x}, \tilde{\mathbf{x}})} \left[\left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} \right\rangle \right] + C_2. \end{aligned} \quad (17)$$

We also have defined in Eq. 9,

$$J_{DSMq_\sigma}(\theta) = \mathbb{E}_{q_\sigma(\mathbf{x}, \tilde{\mathbf{x}})} \left[\frac{1}{2} \left\| \psi(\tilde{\mathbf{x}}; \theta) - \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} \right\|^2 \right],$$

which we can develop as

$$\begin{aligned} J_{DSMq_\sigma}(\theta) &= \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})} \left[\frac{1}{2} \|\psi(\tilde{\mathbf{x}}; \theta)\|^2 \right] \\ &\quad - \mathbb{E}_{q_\sigma(\mathbf{x}, \tilde{\mathbf{x}})} \left[\left\langle \psi(\tilde{\mathbf{x}}; \theta), \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} \right\rangle \right] + C_3 \end{aligned} \quad (18)$$

where $C_3 = \mathbb{E}_{q_\sigma(\mathbf{x}, \tilde{\mathbf{x}})} \left[\frac{1}{2} \left\| \frac{\partial \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}{\partial \tilde{\mathbf{x}}} \right\|^2 \right]$ is a constant that does not depend on θ .

Looking at equations 17 and 18 we see that $J_{ESMq_\sigma}(\theta) = J_{DSMq_\sigma}(\theta) + C_2 - C_3$. We have thus shown that the two optimization objectives are equivalent.

DSM with Additive White Gaussian Noise

- Take the proposal distribution to be

$$\tilde{X} = X + \sigma W \text{ where } W \sim N(0, I)$$

- Then we have that

$$q_{\sigma}(\tilde{x}|x) = \frac{1}{(2\pi\sigma^2)^{\frac{p}{2}}} \exp\left\{-\frac{1}{2\sigma^2} \|\tilde{x} - x\|^2\right\}$$

$$\nabla_{\tilde{x}} \log q_{\sigma}(\tilde{x}|x) = \frac{1}{\sigma^2} (x - \tilde{x})$$

- So, then the DSM loss function is*

$$L_{DSM}(\theta; \sigma) = E \left[\frac{1}{2} \left\| \frac{1}{\sigma^2} (X - \tilde{X}) - s_{\theta}(\tilde{X}) \right\|^2 \right]$$

*noise-less
image*

*noisy
image*

*Score for
distribution of \tilde{X}*

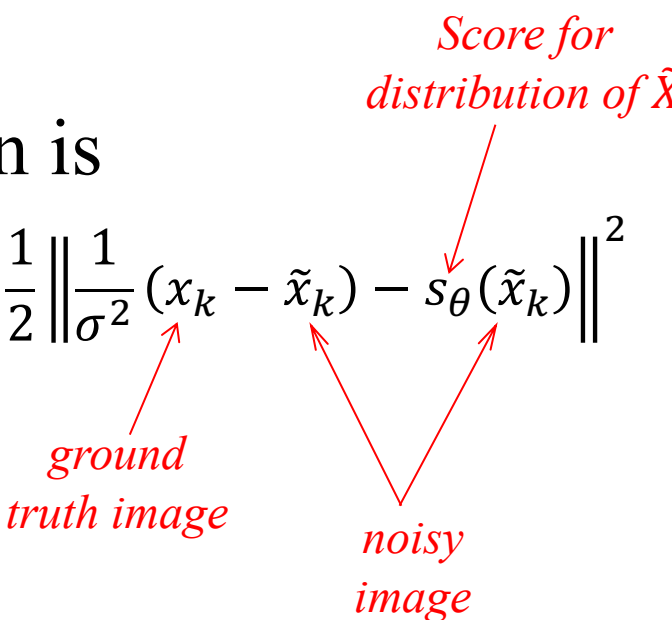
**We can
compute this!**

The DSM with AWGN: Loss Function

- Goal: Formulate loss function from training data
 - $\{x_0, \dots, x_{K-1}\}$ - training samples from desired distribution
 - For $k = 0, \dots, K - 1$, create noisy sample:

$$\tilde{x}_k = x_k + \sigma w_k \text{ where } w \sim N(0, I)$$

- Practical loss function is

$$\theta_\sigma = \arg \min_{\theta} \sum_{k=0}^{K-1} \frac{1}{2} \left\| \frac{1}{\sigma^2} (x_k - \tilde{x}_k) - s_{\theta}(\tilde{x}_k) \right\|^2$$


Score for distribution of \tilde{X}

ground truth image

noisy image

DSM with AWGN: Simplified

- Take the proposal distribution to be

$$\tilde{x}_k = x_k + \sigma w_k \text{ where } w_k \sim N(0, I)$$

- Then we have that

$$L_{DSM}(\theta; \sigma) = \sum_{k=0}^{K-1} \frac{1}{2} \left\| \frac{1}{\sigma^2} (x_k - \tilde{x}_k) - s_{\theta}(\tilde{x}_k) \right\|^2$$

$$= \sum_{k=0}^{K-1} \frac{1}{2} \left\| \frac{w_k}{\sigma} + s_{\theta}(x_k + \sigma w_k) \right\|^2$$

- So then

$$-w_k \approx \sigma s_{\theta_{\sigma}}(x_k + \sigma w_k)$$

Denoising and the Score

- It's easy to show that

$$X = \tilde{X} + \sigma^2 s_{\theta_\sigma}(\tilde{X}) = \text{Denoise}(\tilde{X}; \sigma^2)$$

- or equivalently that

$$s_{\theta_\sigma}(\tilde{X}) = \frac{1}{\sigma^2} [\text{Denoise}(\tilde{X}; \sigma^2) - \tilde{X}]$$

- Interpretation:

- $\text{Denoise}(\tilde{X}; \sigma^2)$ is a MMSE denoiser
- $\sigma s_{\theta_\sigma}(\tilde{X})$ estimates the negative noise.
- This is just residual training for an image denoiser.
- As $\sigma \rightarrow 0$, then $s_{\theta_\sigma}(x) \rightarrow s(x)$

DSM with AWGN: Graphical Interpretation

- Take the proposal distribution to be

$$\tilde{X} = X + \sigma W \text{ where } W \sim N(0, I)$$


- If we first define

$$\begin{aligned}\tilde{L}_{DSM}(\theta, \tilde{x}; \sigma) &= E \left[\frac{1}{2} \left\| \frac{1}{\sigma^2} (X - \tilde{x}) - s_{\theta}(\tilde{x}) \right\|^2 \middle| \tilde{X} = \tilde{x} \right] \\ &= \int_{\mathbb{R}^N} \frac{1}{2} \left\| \frac{1}{\sigma^2} (x - \tilde{x}) - s_{\theta}(\tilde{x}) \right\|^2 p_{\sigma^2}(x|\tilde{x}) dx\end{aligned}$$

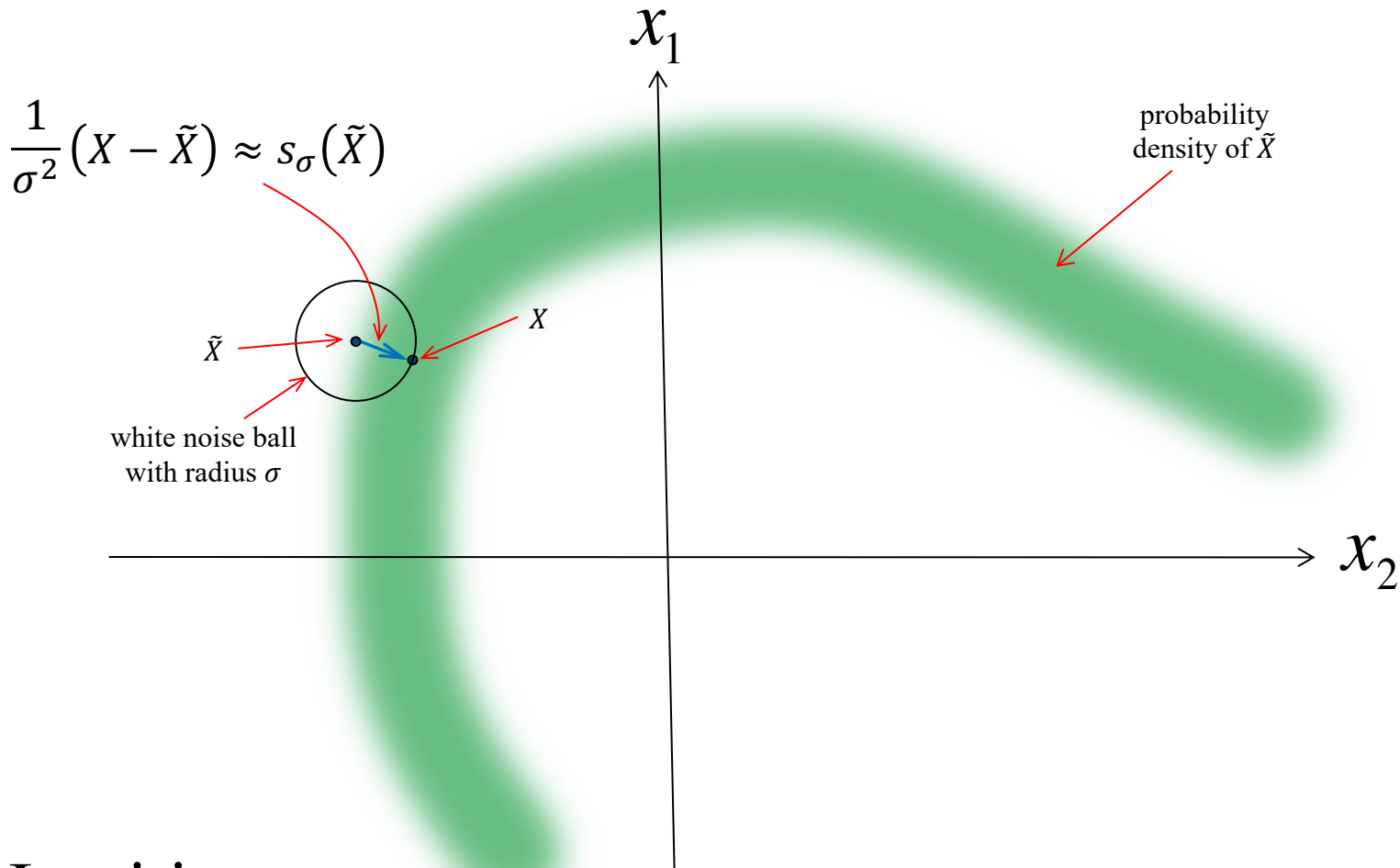
- Then we have that

$$\begin{aligned}L_{DSM}(\theta; \sigma) &= E[\tilde{L}_{DSM}(\theta, \tilde{X}; \sigma)] \\ &= \int_{\mathbb{R}^N} \tilde{L}_{DSM}(\theta, \tilde{x}; \sigma) p_{\sigma^2}(\tilde{x}) d\tilde{x}\end{aligned}$$

*Posterior distribution
of noiseless image
given noisy image*



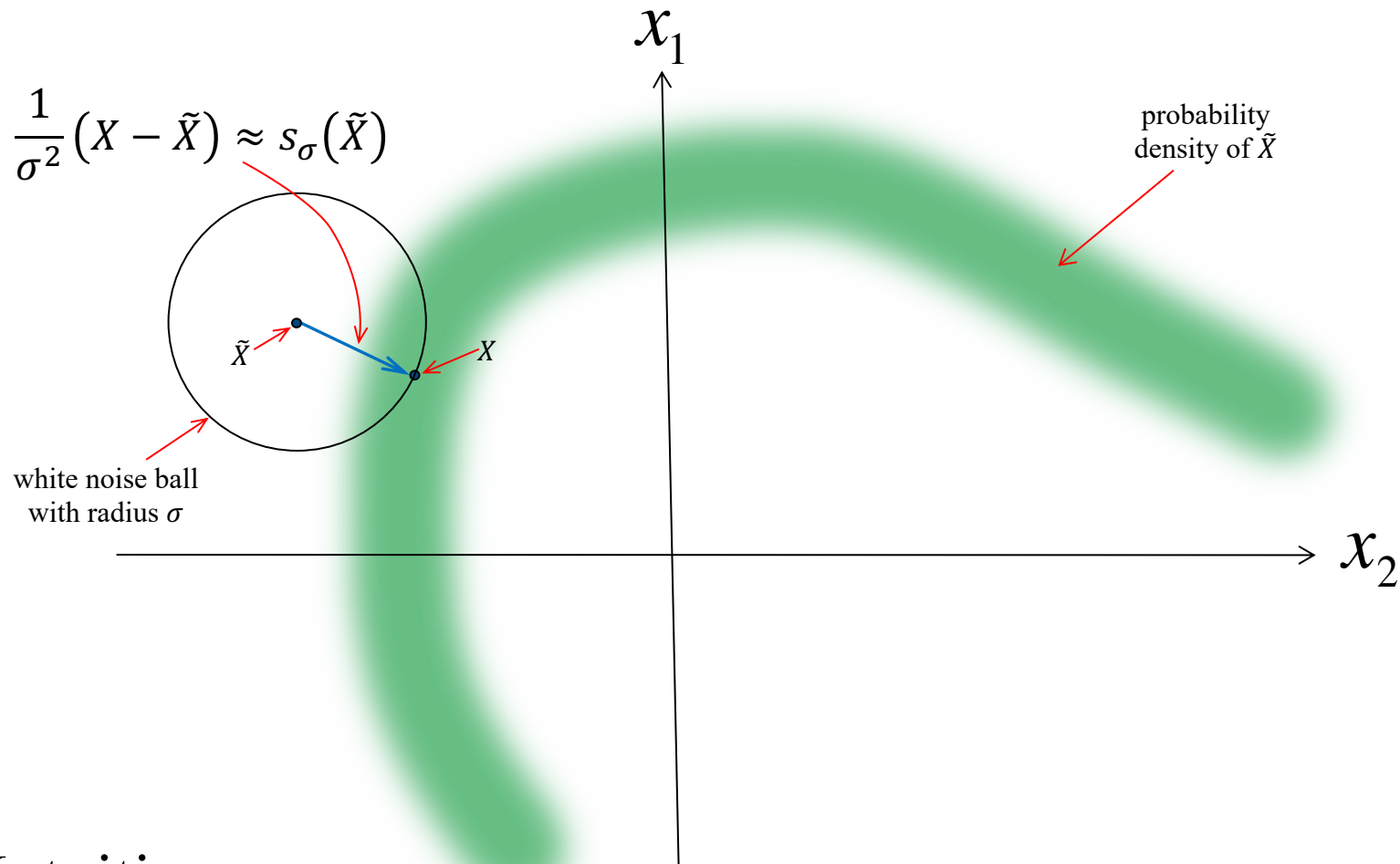
Interpretation of Denoising Score Matching



■ Intuition:

- Denoiser moves towards larger probability
- Expected change approximates score

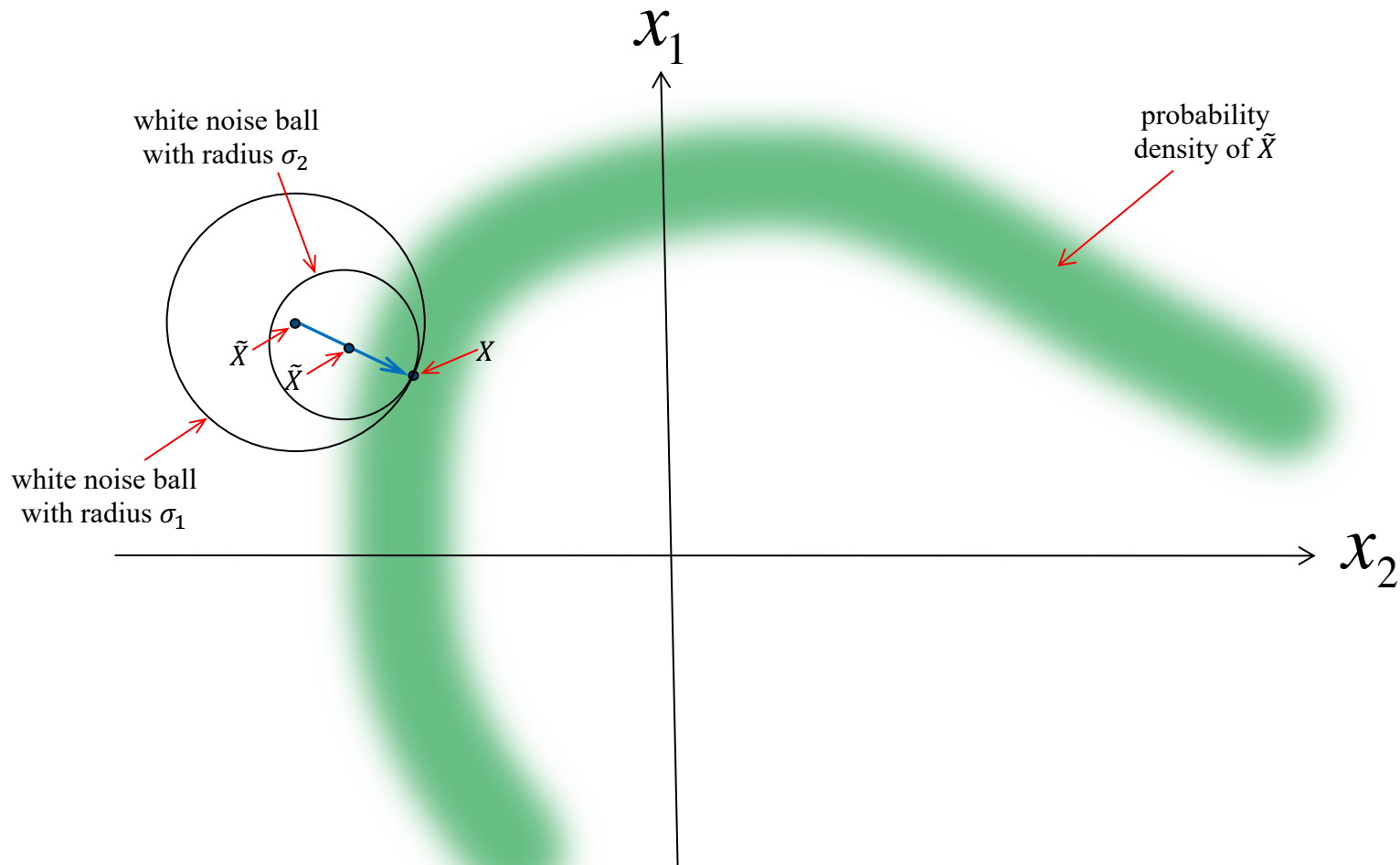
Interpretation of DSM with larger σ



■ Intuition:

- Samples further from the peak of the distribution
- Allows for sample in low probability regions
- Speeds convergence of MCMC

DSM with Decreasing σ



■ Intuition:

- Large σ samples far from the peak \Rightarrow used early in the simulation
- Small σ samples close to the peak \Rightarrow used late in the simulation

Generative Diffusion Models^{*†}

- Langevin dynamics

^{*}[Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole, “Score-Based Generative Modeling Through Stochastic Differential Equations” ICLR 2021.](#)

[†]Yang Song, “Generative Modeling by Estimating Gradients of the Data Distribution,” web blog post, May 5, 2021, <https://yang-song.net/blog/2021/score> .

Langevin Dynamics*

- How can you use the score to generate samples from the Gibbs distribution?
- Langevin dynamics:

$$X_n = X_{n-1} + \epsilon \nabla_x u(X_{n-1}) + \sqrt{2\epsilon} W_n$$

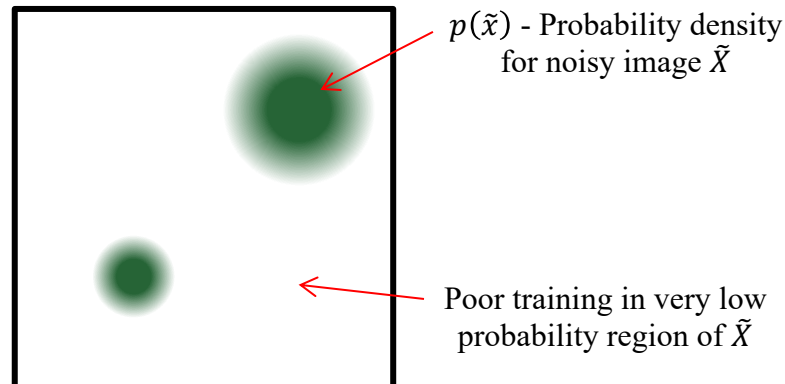
- We can use our estimate of the score to generate

$$X_n = X_{n-1} + \epsilon s_{\theta_\sigma}(X_{n-1}) + \sqrt{2\epsilon} W_n$$

*Score learns the gradient of
the log probability.*

*White noise,
 $W_n \sim N(0, I)$.*

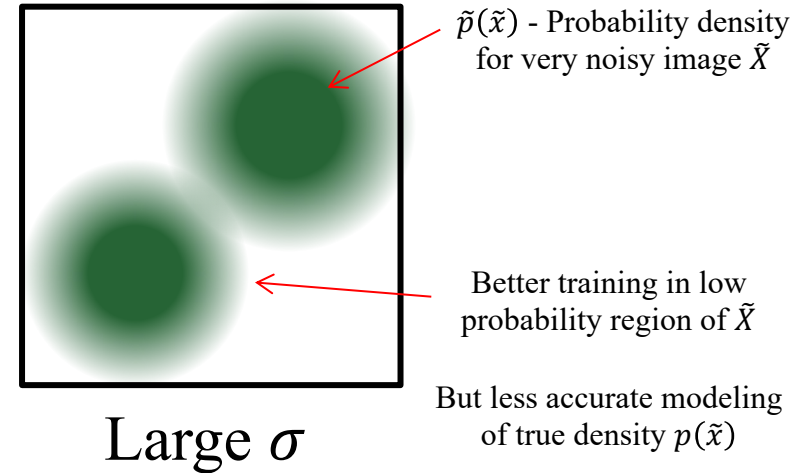
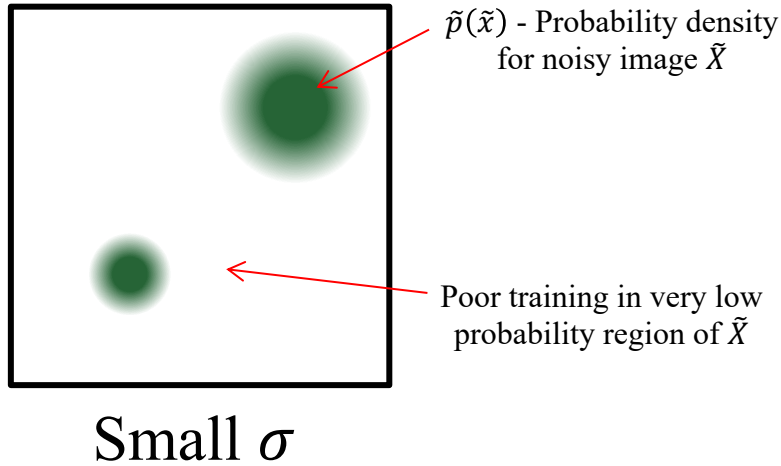
- Problem: Takes too long to converge



Annealed Langevin Dynamics*

- Key idea: Increase σ to get better estimation in low density regions

- Small vs Large values of σ



But less accurate modeling of true density $p(\tilde{x})$

- Annealed Langevin dynamics:

- Pick ϵ_o and let $\sigma_1 > \sigma_2 > \dots > \sigma_N$

For $n = 1$ to N {

$$\epsilon_n \leftarrow \epsilon_o \frac{\sigma_n}{\sigma_L}$$

$$X_n \leftarrow X_{n-1} + \epsilon_n s_{\theta_{\sigma_n}}(X_{n-1}) + \sqrt{2\epsilon_n} W_n$$

}

Practical Recommendations: Annealed*†

■ Annealed Langevin dynamics:

- Pick ϵ_o and let $\sigma_1 > \sigma_2 > \dots > \sigma_N$

$\epsilon_o \leftarrow \text{init}; \sigma_{\min} \leftarrow \text{init}; \sigma_{\max} \leftarrow \text{init};$

$\alpha \leftarrow \left(\frac{\sigma_{\min}}{\sigma_{\max}} \right)^{\frac{1}{N-1}};$

For $n = 0$ to $N - 1$ {

$\sigma_n \leftarrow \alpha^n \sigma_{\max}$

$\epsilon_n \leftarrow \epsilon_o \frac{\sigma_n}{\sigma_{\max}}$

$X_n \leftarrow X_{n-1} + \epsilon_n s_{\theta_{\sigma_n}}(X_{n-1}) + \sqrt{2\epsilon_n} W_n$

}

Annealed Langevin Dynamics

■ Practical considerations

- Geometric sequence for σ_n
- $\sigma_{\max} = \max_{i,j} \text{RMS}(X_i - X_j)$ where X_i and X_j are training images.
- Use a U-net (RefineNet) with skipped connections for score modeling.
- Apply exponential moving average on the weights of the score-based model when used at test time.

*Yang Song, “Generative Modeling by Estimating Gradients of the Data Distribution,” web blog post, May 5, 2021, <https://yang-song.net/blog/2021/score>.

†Yang Song, Y. and Stefan Ermon, “Improved Techniques for Training Score-Based Generative Models”, Neural Information Processing Systems 2020.

Langevin: Denoising Interpretation

- Annealed Langevin dynamics:

$$X_n = X_{n-1} + \epsilon_n s_{\theta_{\sigma_n}}(X_{n-1}) + \sqrt{2\epsilon_n} W_n$$

- where

$$s_{\theta_\sigma}(x) = \frac{1}{\sigma^2} [\text{Denoise}(x; \sigma^2) - x]$$

- If we set $\epsilon_n = \sigma^2$, then we get

$$X_n = \text{Denoise}(X_{n-1}; \sigma^2) + \sqrt{2}\sigma W_n$$

- where $W_n \sim N(0, I)$

- Interpretation:

- Remove noise with variance σ^2 , then add AWGN with variance $2\sigma^2$.
 - As $\sigma \rightarrow 0$, this iteration generates samples from the distribution $p(x)$.

Denoising Interpretation of Langevin

■ Annealed Langevin dynamics:

$\sigma_{\min} \leftarrow \text{init}; \sigma_{\max} \leftarrow \text{init};$

$\alpha \leftarrow \left(\frac{\sigma_{\min}}{\sigma_{\max}} \right)^{\frac{1}{N-1}};$

For $n = 0$ to $N - 1$ {

$\sigma_n \leftarrow \alpha^n \sigma_{\max}$

$X_n \leftarrow \text{Denoise}(X_{n-1}; \sigma_n^2) + \sqrt{2}\sigma_n W_n$

}

*Annealed Langevin Dynamics:
Denoising Interpretation*

• Interpretation:

- Remove noise with variance σ^2 , then add back AWGN with variance $2\sigma^2$.
- Denoiser trained using MMSE loss on samples from $p(x)$ with AWGN of variance σ^2 .
- As $\sigma \rightarrow 0$, this iteration generates samples from the distribution $p(x)$.