Generative Modeling by Estimating Gradients of the Data Distribution

YANG SONG

Progress in generative models of images

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J. and Aila, T., 2020. Analyzing and improving the image quality of stylegan. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8110-8119).

Progress in generative models of text

Progress in generative models of text

Talktotransformer.com

(prompt) The best generative model is

Progress in generative models of text

Talktotransformer.com

(prompt) The best generative model is

one that can learn over time and which predicts the structure and functionality of the brain as a whole.

Introduction to generative models

Introduction to generative models

$$\mathbf{x}_i \overset{\text{i.i.d.}}{\sim} p_{\text{data}}$$

$$i = 1, 2, \cdots, N$$

Introduction to generative models

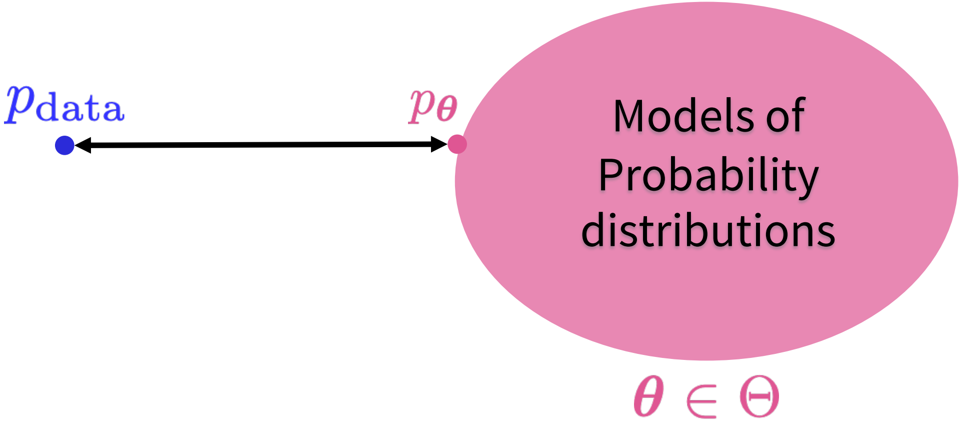
$$\mathbf{x}_i \overset{\text{i.i.d.}}{\sim} p_{\text{data}}$$
$$i = 1, 2, \cdots, N$$

Models of Probability distributions

$\boldsymbol{\theta} \in \Theta$

Introduction to generative models

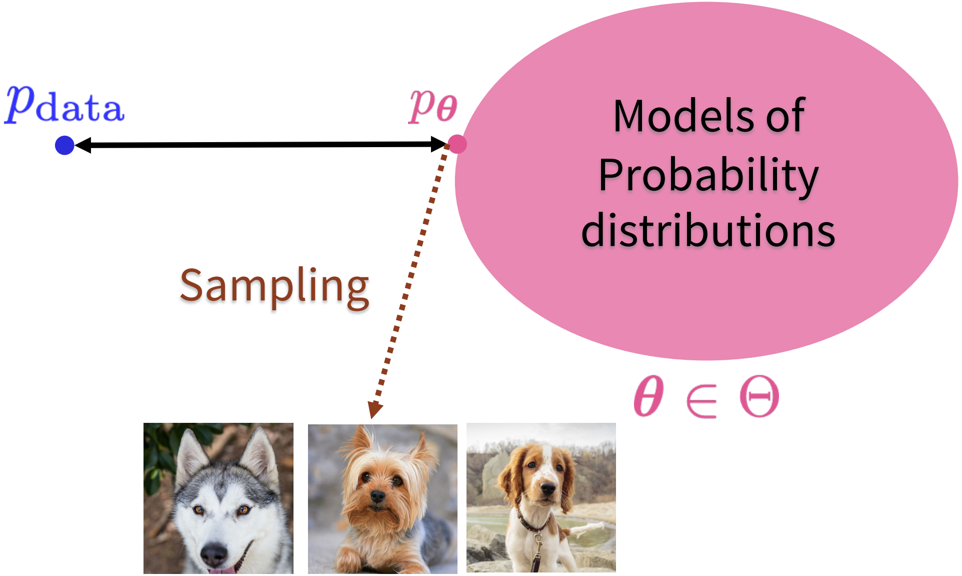
$$\mathbf{x}_i \overset{\text{i.i.d.}}{\sim} p_{\text{data}}$$
$$i = 1, 2, \cdots, N$$



Introduction to generative models

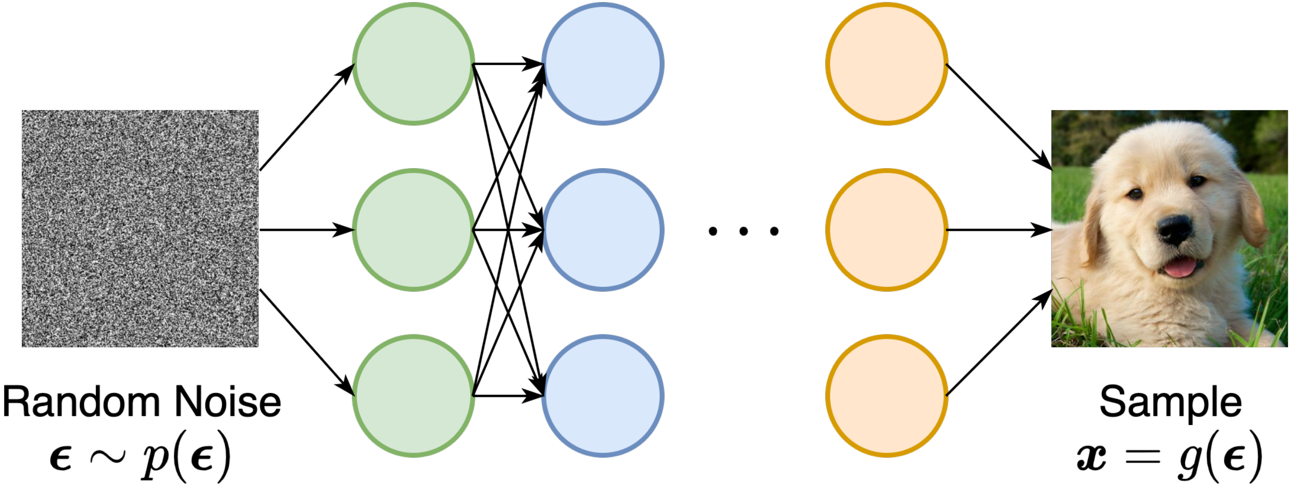
$$\mathbf{x}_i \overset{\text{i.i.d.}}{\sim} p_{\text{data}}$$

$$i = 1, 2, \cdots, N$$



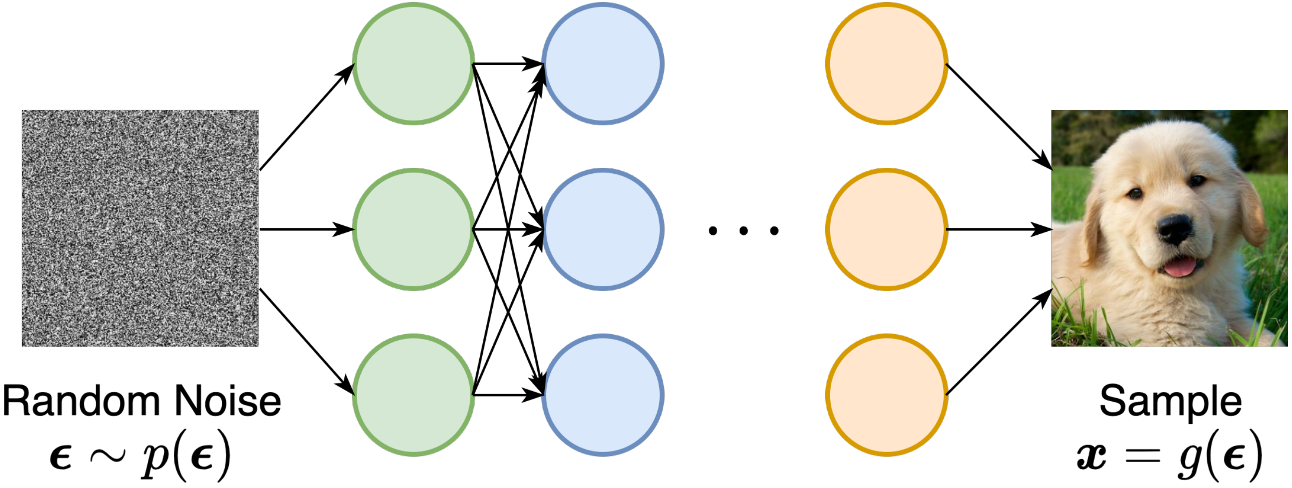
Representations of probability distributions

Implicit models: represent the sampling process



Representations of probability distributions

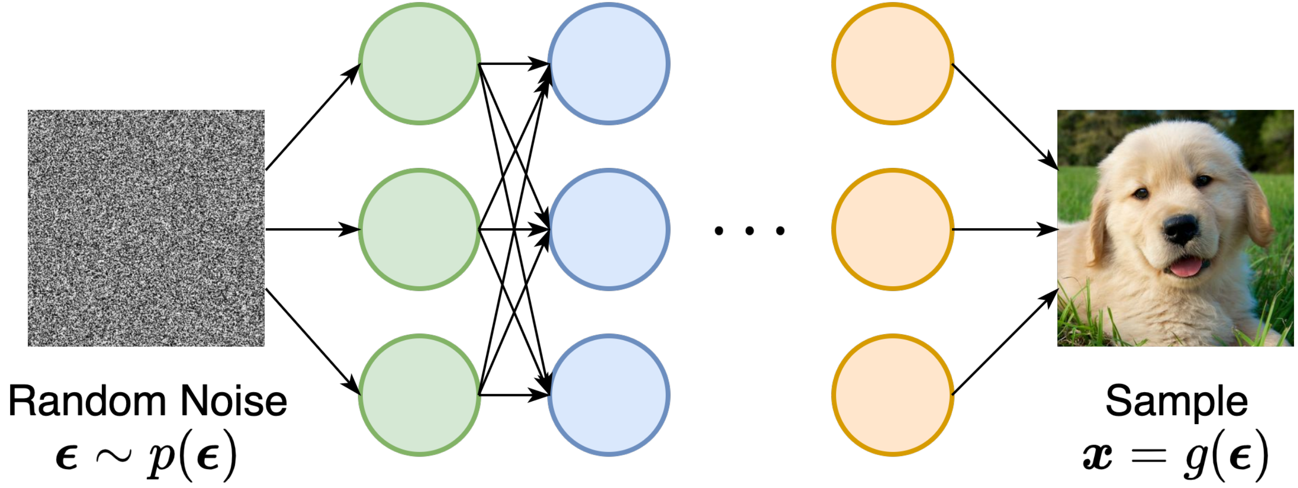
Implicit models: represent the sampling process



Pros: flexible architecture, high sample quality.

Representations of probability distributions

Implicit models: represent the sampling process

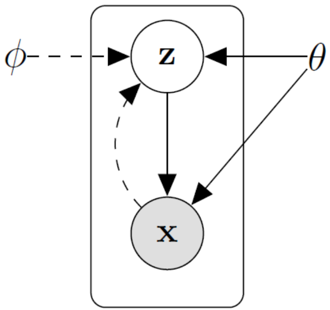


Pros: flexible architecture, high sample quality.

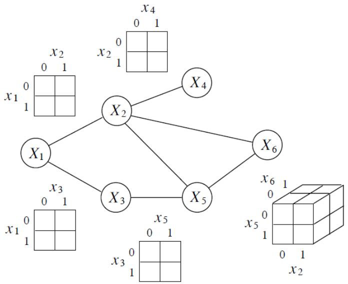
Cons: hard to train, no likelihood, no principled model comparisons

Representations of probability distributions

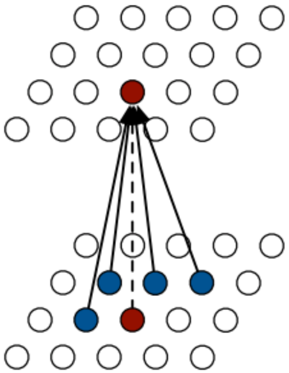
Explicit models: represent a probability density/mass function



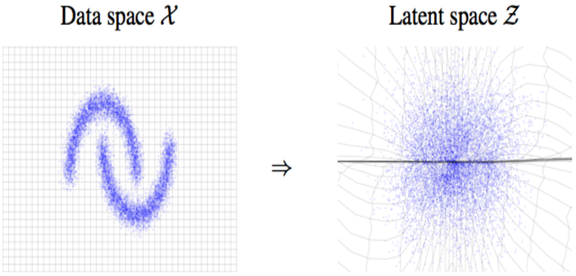
Bayesian networks (e.g., VAEs)



MRF



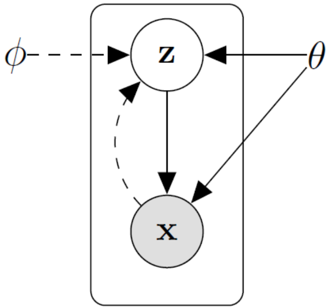
Autoregressive models



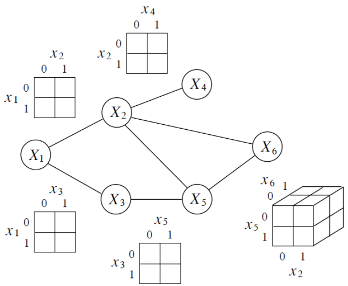
Flow models

Representations of probability distributions

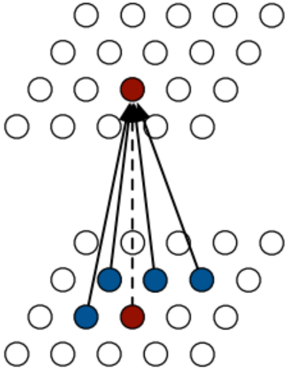
Explicit models: represent a probability density/mass function



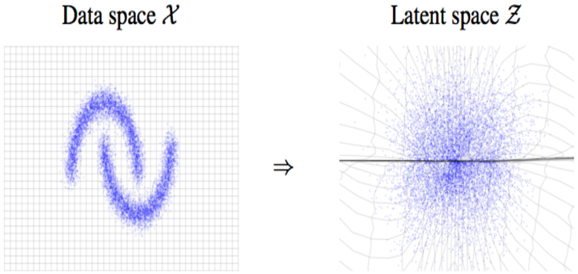
Bayesian networks (e.g., VAEs)



MRF



Autoregressive models

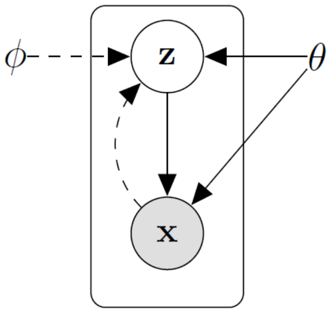


Flow models

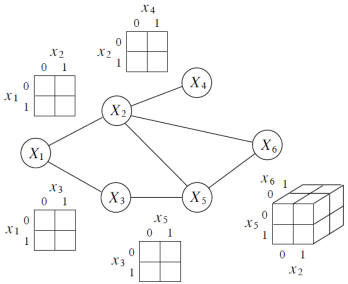
Pros: likelihoods

Representations of probability distributions

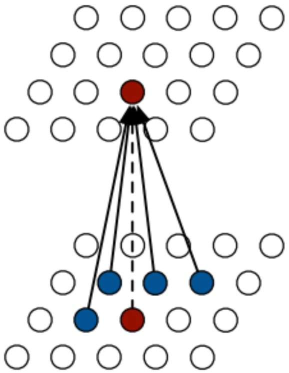
Explicit models: represent a probability density/mass function



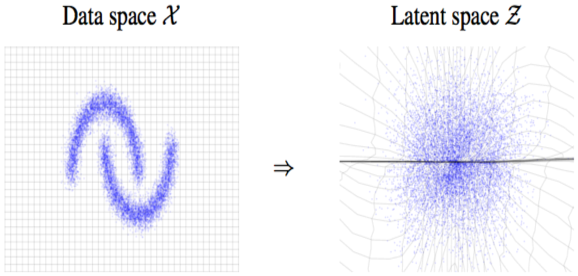
Bayesian networks (e.g., VAEs)



MRF



Autoregressive models



Flow models

Pros: likelihoods

Cons: need to be normalized, expressivity-tractability trade-off

Representation of probability distributions

This talk: The gradient of log probability density

Representation of probability distributions

This talk: The gradient of log probability density

$$\nabla_{\mathbf{x}} \log p(\mathbf{x})$$

Representation of probability distributions

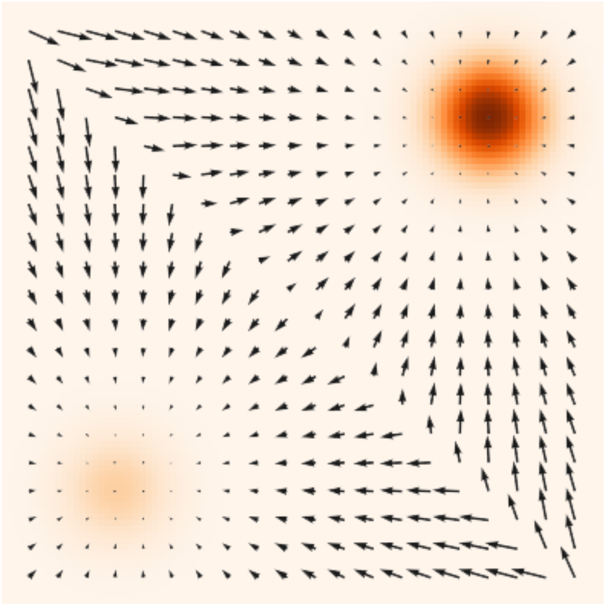
This talk: The gradient of log probability density

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) \quad \text{Score}$$

Representation of probability distributions

This talk: The gradient of log probability density

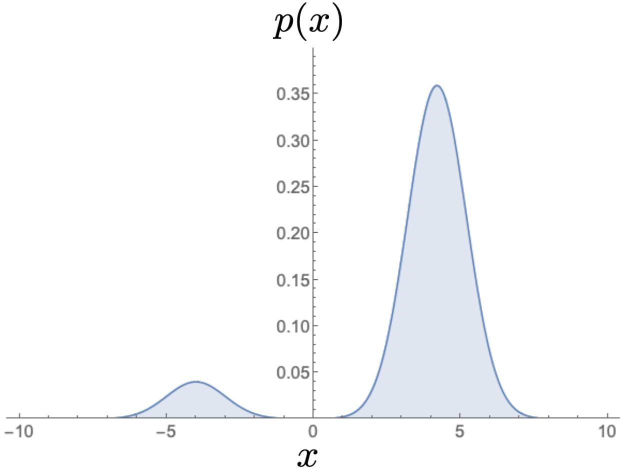
$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) \quad \text{Score}$$



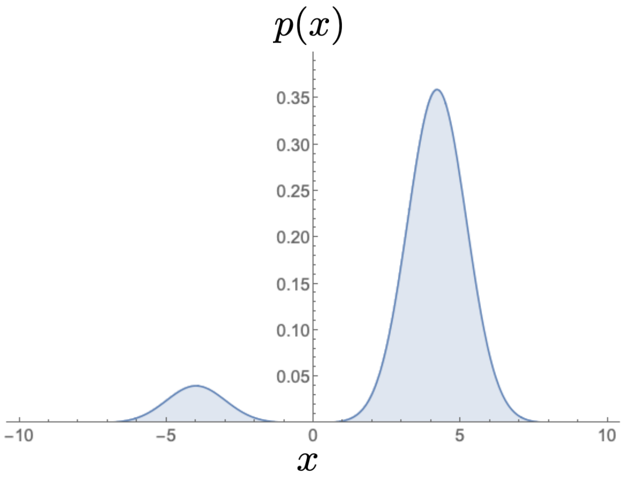
(PDF and score)

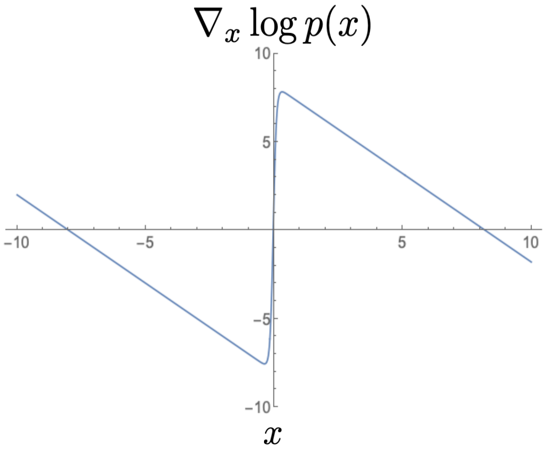
Why scores?

Why scores?

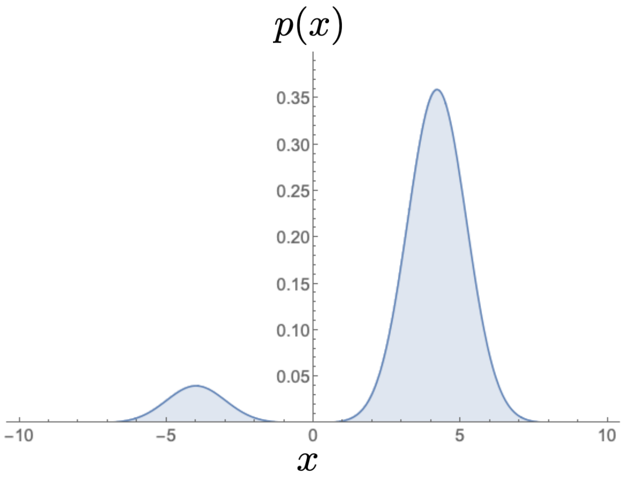


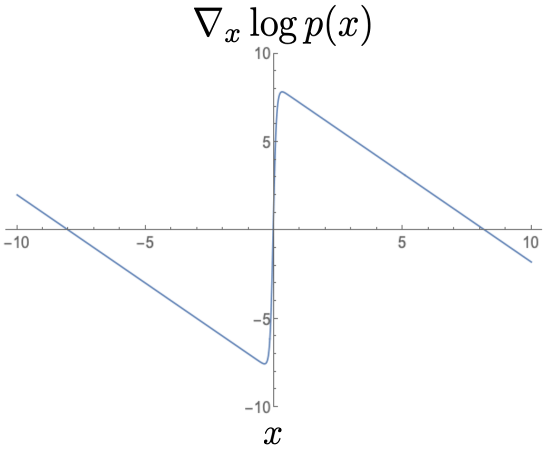
Why scores?

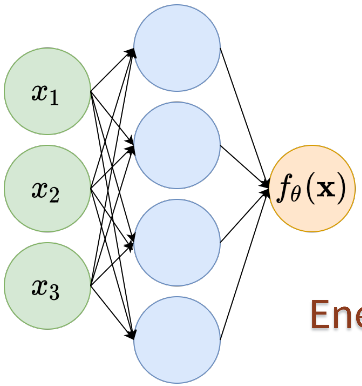

$p(x)$
0.35
0.30
0.25
0.20
0.15
0.10
0.05
−10
−5
0
5
10
x


$\nabla_x \log p(x)$
10
5
−5
−10
−10
−5
5
10
x

Why scores?



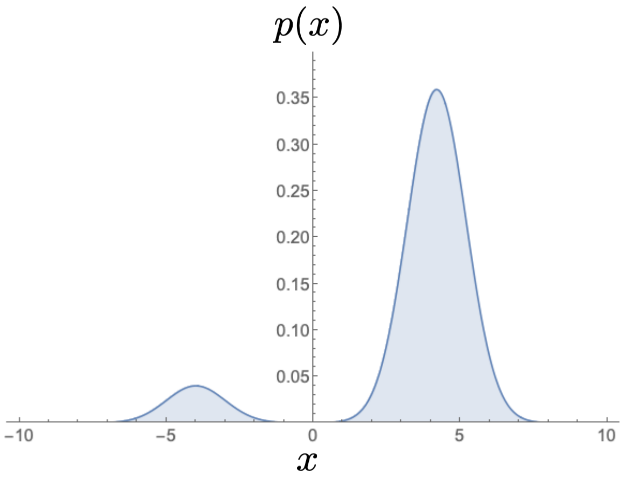


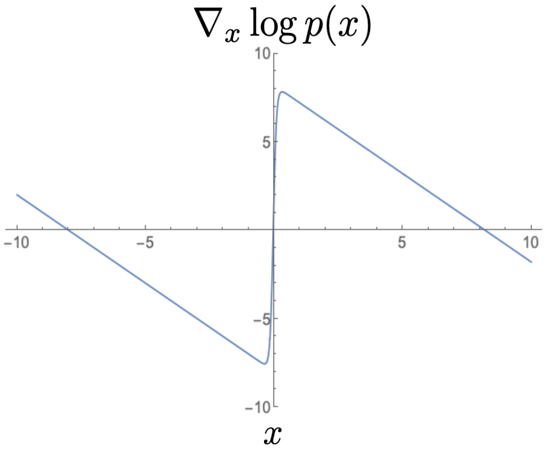


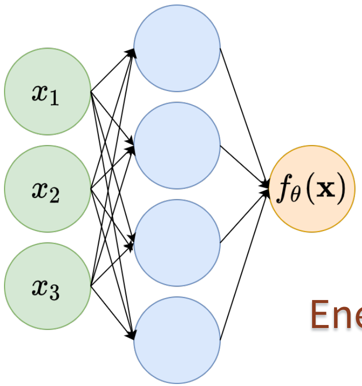
$$\Longrightarrow p_\theta(\mathbf{x}) = \frac{e^{-f_\theta(\mathbf{x})}}{Z_\theta}$$

Energy-Based Model (EBM)

Why scores?







$$\Longrightarrow p_\theta(\mathbf{x}) = \frac{e^{-f_\theta(\mathbf{x})}}{Z_\theta}$$

Energy-Based Model (EBM)

$$\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z_\theta}_{=0}$$
$$= -\nabla_{\mathbf{x}} f_\theta(\mathbf{x})$$

Score

Score estimation

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$
- **Goal:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Score estimation

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \stackrel{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$
- **Goal:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- **Score Model:** A trainable vector-valued function $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^D$

Score estimation

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$
- **Goal:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- **Score Model:** A trainable vector-valued function $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^D$
- **Objective:** How to compare two vector fields of scores?

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$

(Fisher divergence)

Score estimation

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$
- **Goal:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- **Score Model:** A trainable vector-valued function $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^D$
- **Objective:** How to compare two vector fields of scores?

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$

?

(Fisher divergence)

Score estimation

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$
- **Goal:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- **Score Model:** A trainable vector-valued function $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^D$
- **Objective:** How to compare two vector fields of scores?

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$

?

(Fisher divergence)

- Integration by parts → **Score Matching (**Hyvärinen 2005)

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2 + \text{trace}(\underbrace{\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})}_{\text{Jacobian of } \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})})\right]$$

Score estimation

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$
- **Goal:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- **Score Model:** A trainable vector-valued function $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^D$
- **Objective:** How to compare two vector fields of scores?

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$

?

(Fisher divergence)

- Integration by parts → **Score Matching (**Hyvärinen 2005)

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2 + \operatorname{trace}(\underbrace{\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})}_{\text{Jacobian of } \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})})\right]$$

$$\approx \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_i)\|_2^2 + \operatorname{trace}(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_i))\right]$$

Score estimation

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$
- **Goal:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- **Score Model:** A trainable vector-valued function $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^D$
- **Objective:** How to compare two vector fields of scores?

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$

?

(Fisher divergence)

- Integration by parts → **Score Matching (**Hyvärinen 2005)

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2 + \boxed{\text{trace}(\underbrace{\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})}_{\text{Jacobian of } \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})})}\right]$$

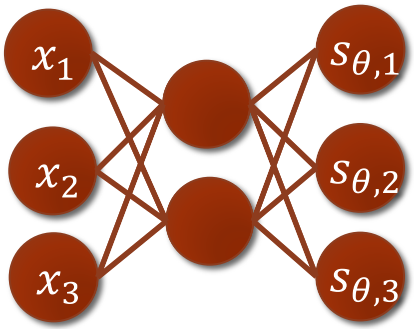
$$\approx \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_i)\|_2^2 + \boxed{\text{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_i))}\right]$$

Not scalable

Stanford University

Score Matching is not scalable

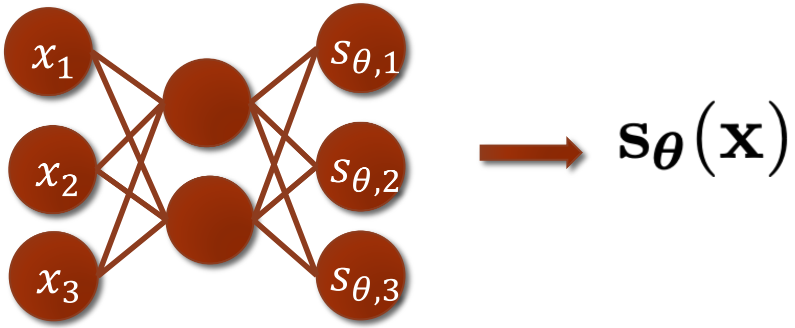
- Deep score models



$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

Score Matching is not scalable

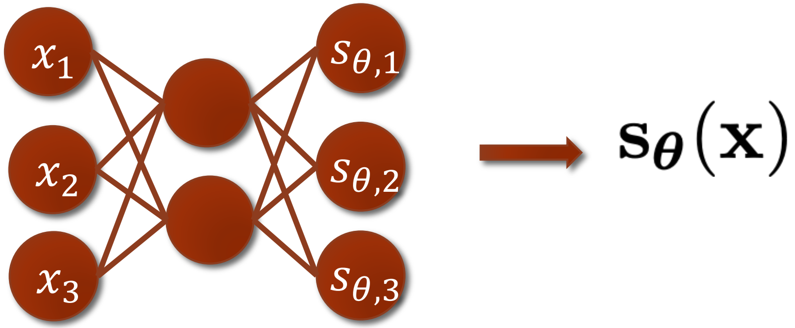
- Deep score models



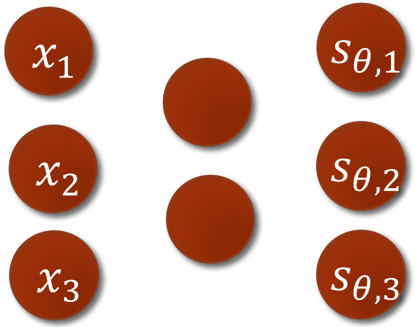
- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

Score Matching is not scalable

- Deep score models

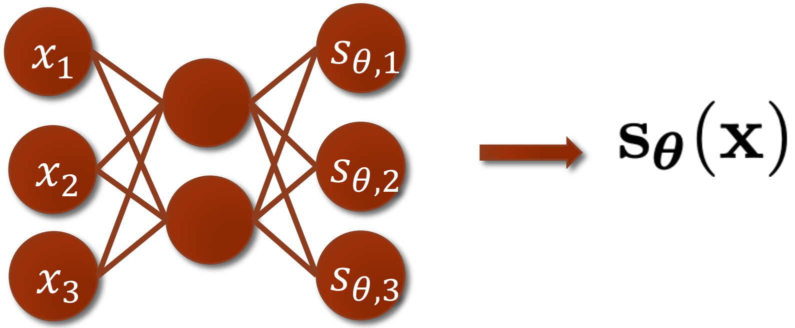


- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

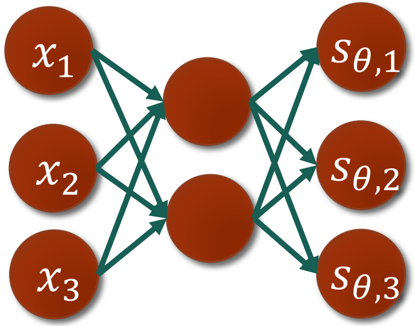


Score Matching is not scalable

- Deep score models

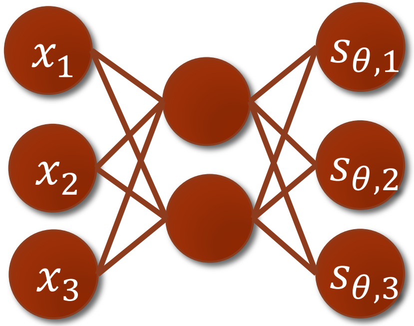


- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$



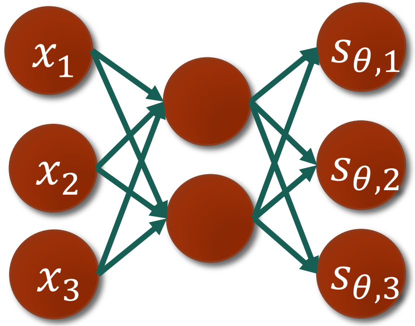
Score Matching is not scalable

- Deep score models



$\rightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

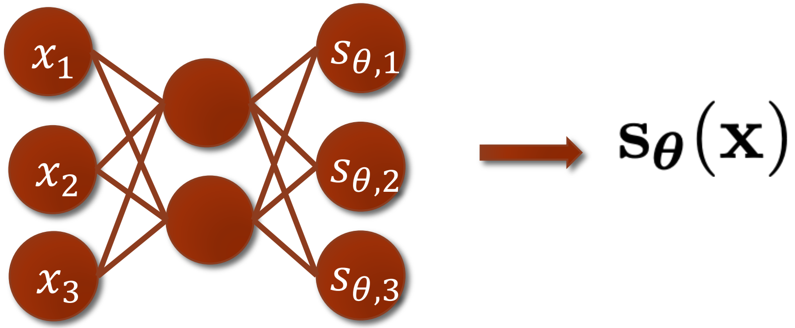
- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$



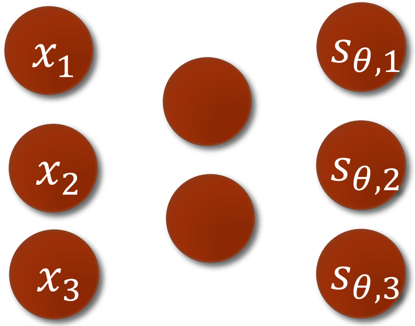
$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

Score Matching is not scalable

- Deep score models

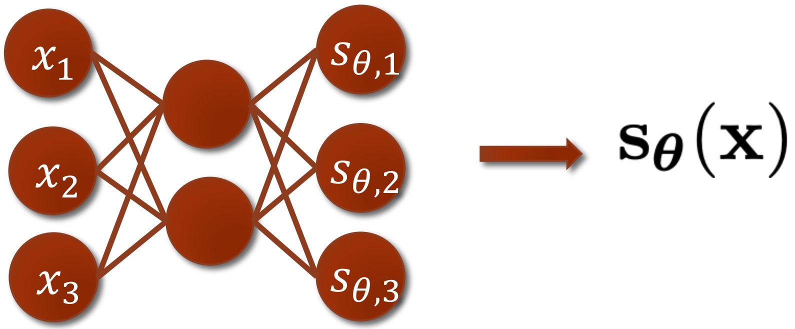


- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

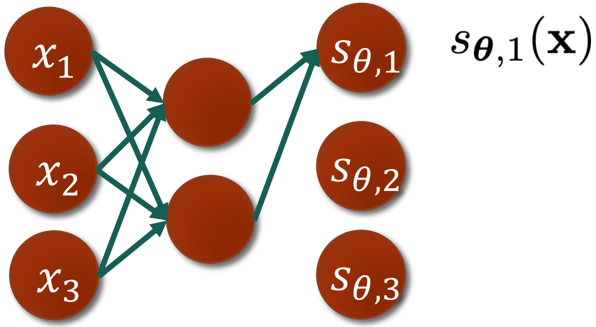


Score Matching is not scalable

- Deep score models

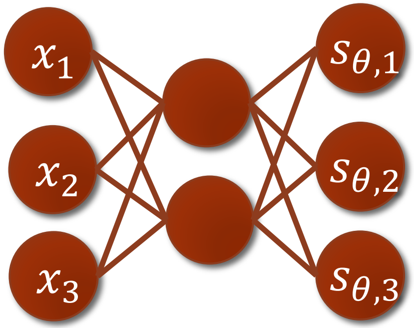


- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$



Score Matching is not scalable

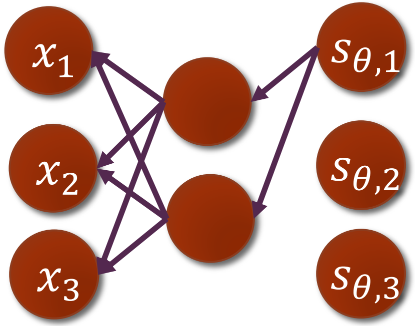
- Deep score models



$$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$$

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

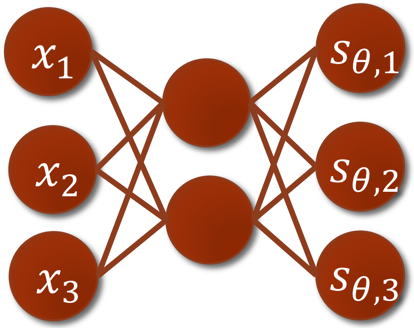
$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$



$$\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3} \end{pmatrix}$$

Score Matching is not scalable

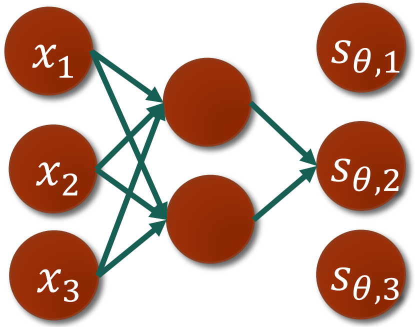
- Deep score models



$$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$$

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$

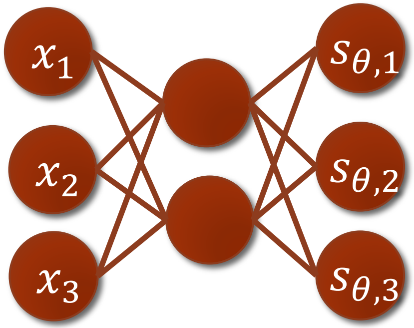


$$s_{\boldsymbol{\theta},2}(\mathbf{x})$$

$$\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3} \end{pmatrix}$$

Score Matching is not scalable

- Deep score models

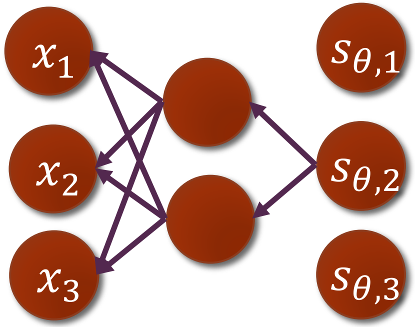


$$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$$

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\mathrm{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$

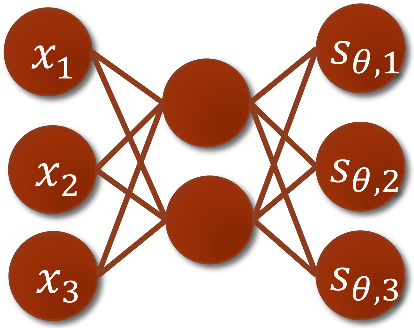
$$\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}$$



$$\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \boxed{\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3} \end{pmatrix}$$

Score Matching is not scalable

- Deep score models

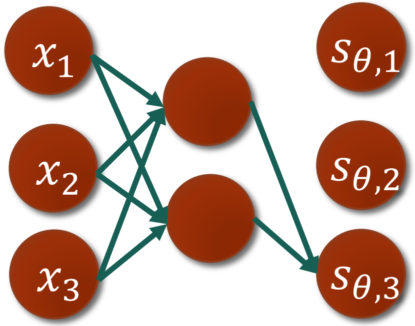


$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$

$$\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}$$

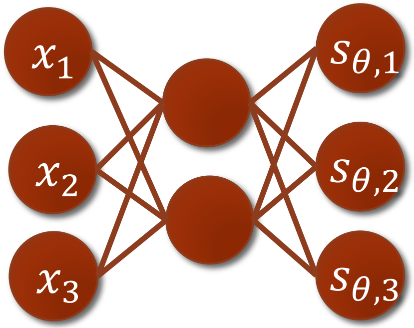


$s_{\boldsymbol{\theta},3}(\mathbf{x})$

$$\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \boxed{\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3} \end{pmatrix}$$

Score Matching is not scalable

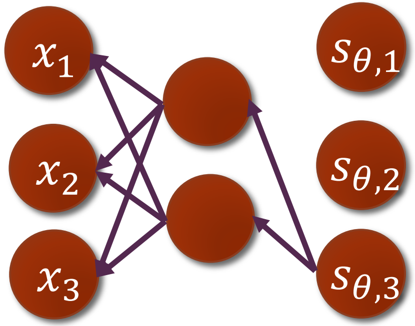
- Deep score models



$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

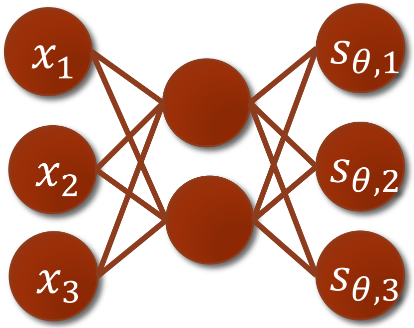
$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$
$$\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}$$
$$\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}$$



$$\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \boxed{\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \boxed{\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}} \end{pmatrix}$$

Score Matching is not scalable

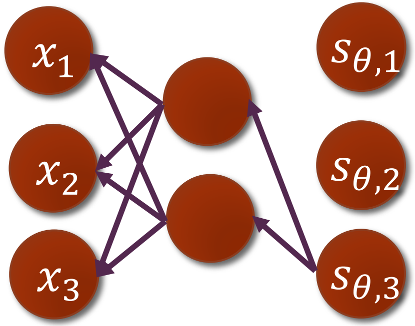
- Deep score models



$$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$$

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$
$$\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}$$
$$\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}$$

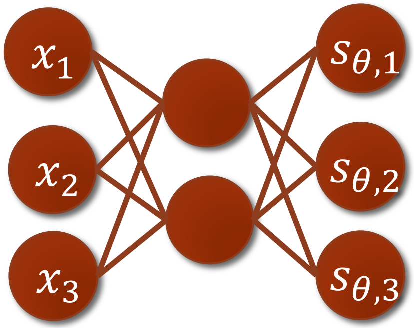


$O(D)$ Backprops!

$$\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \boxed{\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \boxed{\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}} \end{pmatrix}$$

Score Matching is not scalable

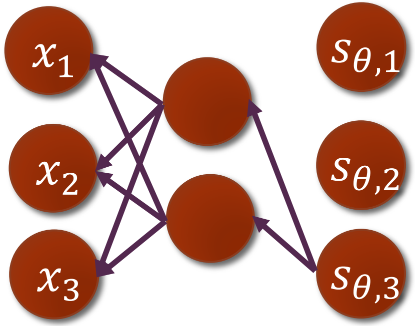
- Deep score models



$$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$$

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\nabla_{\mathbf{x}} \cancel{\mathbf{s}_{\boldsymbol{\theta}}}(\mathbf{x}))$

$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$
$$\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}$$
$$\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}$$

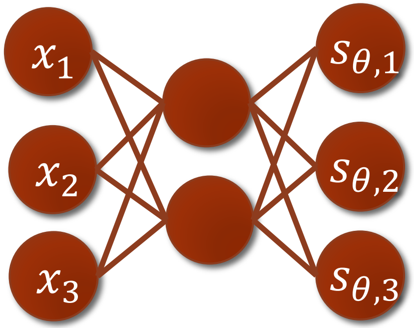


$O(D)$ Backprops!

$$\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \boxed{\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \boxed{\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}} \end{pmatrix}$$

Score Matching is not scalable

- Deep score models

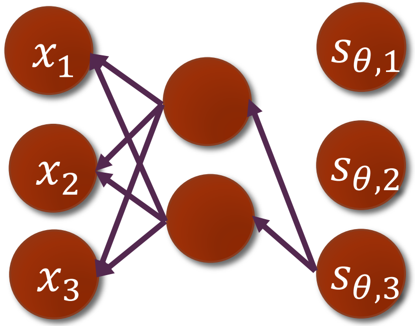


$\longrightarrow \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

Score Matching is not Scalable!

- Compute $\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2$ and $\operatorname{trace}(\cancel{\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}}(\mathbf{x}))$

$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$
$$\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}$$
$$\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}$$



$O(D)$ Backprops!

$$\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{pmatrix} \boxed{\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_1} & \boxed{\frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_2}} & \frac{\partial s_{\boldsymbol{\theta},2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_2} & \boxed{\frac{\partial s_{\boldsymbol{\theta},3}(\mathbf{x})}{\partial x_3}} \end{pmatrix}$$

Sliced score matching

- **Intuition:** one dimensional problems should be easier

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Sliced score matching

- **Intuition:** one dimensional problems should be easier
- **Idea:** project onto random directions

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Sliced score matching

- **Intuition:** one dimensional problems should be easier
- **Idea:** project onto random directions
- **Randomized objective: Sliced Fisher Divergence**

$$\frac{1}{2}\mathbb{E}_{p_{\mathbf{v}}}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\left(\mathbf{v}^{\intercal}\nabla_{\mathbf{x}}\log p_{\text{data}}(\mathbf{x}) - \mathbf{v}^{\intercal}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right)^2\right]$$

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Sliced score matching

- **Intuition:** one dimensional problems should be easier
- **Idea:** project onto random directions
- **Randomized objective: Sliced Fisher Divergence**

$$\frac{1}{2}\mathbb{E}_{p_{\mathbf{v}}}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\left(\mathbf{v}^\intercal\nabla_{\mathbf{x}}\log p_{\text{data}}(\mathbf{x}) - \mathbf{v}^\intercal\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right)^2\right]$$

- Integration by parts → **Sliced Score Matching**:

$$\mathbb{E}_{p_{\mathbf{v}}}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\mathbf{v}^\intercal\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v} + \frac{1}{2}\left(\mathbf{v}^\intercal\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right)^2\right]$$

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Sliced score matching

- **Intuition:** one dimensional problems should be easier
- **Idea:** project onto random directions
- **Randomized objective: Sliced Fisher Divergence**

$$\frac{1}{2}\mathbb{E}_{p_{\mathbf{v}}}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[(\mathbf{v}^\intercal\nabla_{\mathbf{x}}\log p_{\text{data}}(\mathbf{x}) - \mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))^2]$$

- Integration by parts → **Sliced Score Matching**:

$$\mathbb{E}_{p_{\mathbf{v}}}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\left[\underbrace{\boxed{\mathbf{v}^\intercal\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v}}}_{\text{Scalable!}} + \frac{1}{2}(\mathbf{v}^\intercal\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))^2\right]$$

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

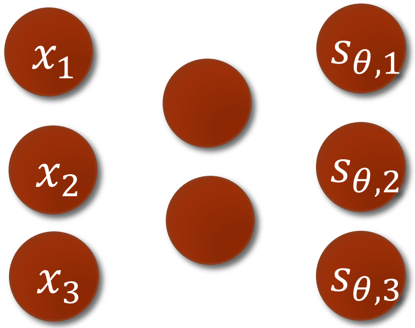
Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$$

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

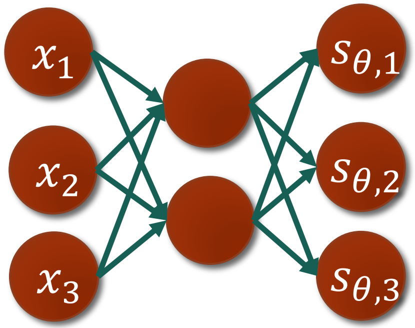
$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$$



Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

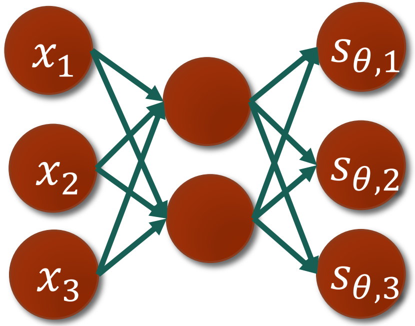
$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$$



Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \boxed{\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})})$$

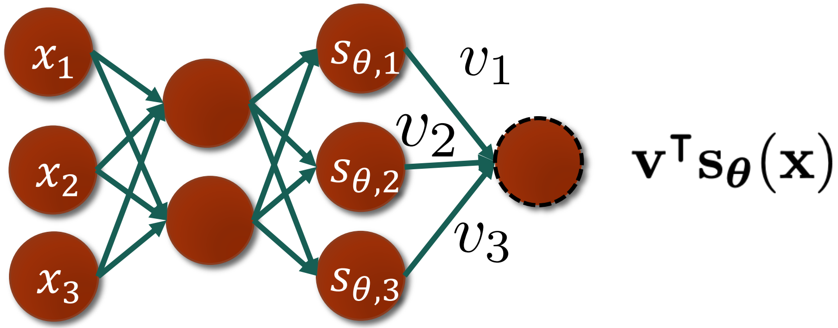


$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} \left(\boxed{\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})} \right)$$

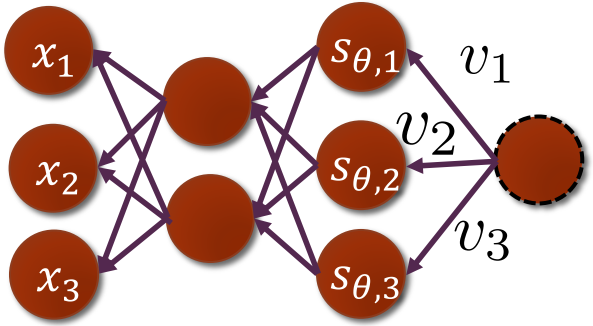


Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v} = \mathbf{v}^\intercal \boxed{\nabla_{\mathbf{x}}(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))}$$

$\nabla_{\mathbf{x}}(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

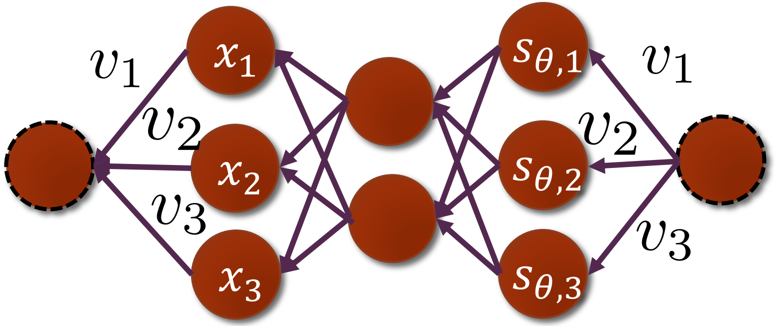


Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v} = \boxed{\mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))}$$

$\mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$



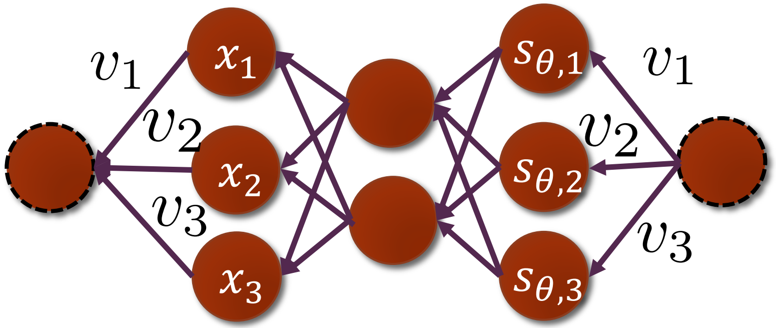
Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \boxed{\mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))}$$

One Backprop!

$\mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$

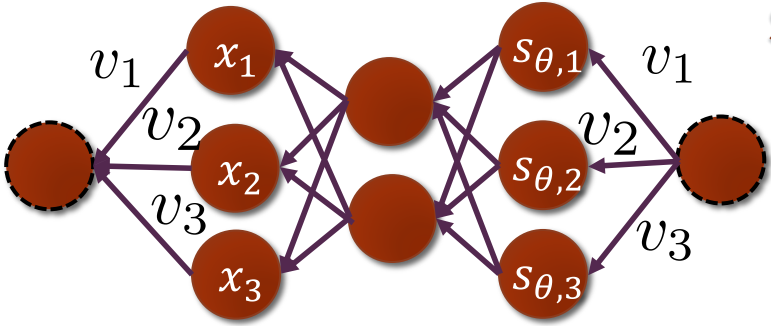


Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \boxed{\mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))}$$

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$$



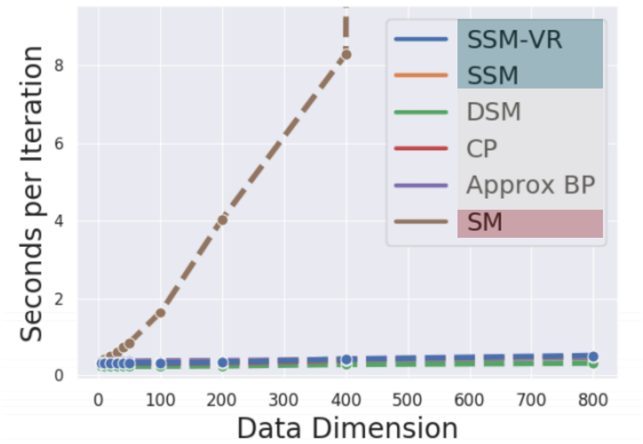
One Backprop!

Sliced Score Matching is scalable

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Results: Sliced Score Matching for EBMs

Sliced score matching methods	Other Baselines	Score Matching

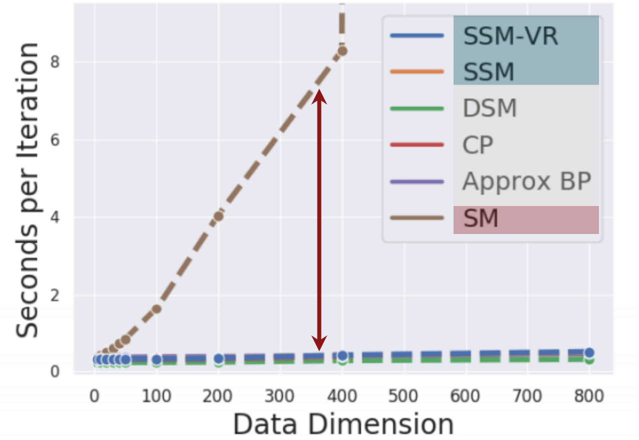


Efficiency

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Results: Sliced Score Matching for EBMs

Sliced score matching methods	Other Baselines	Score Matching

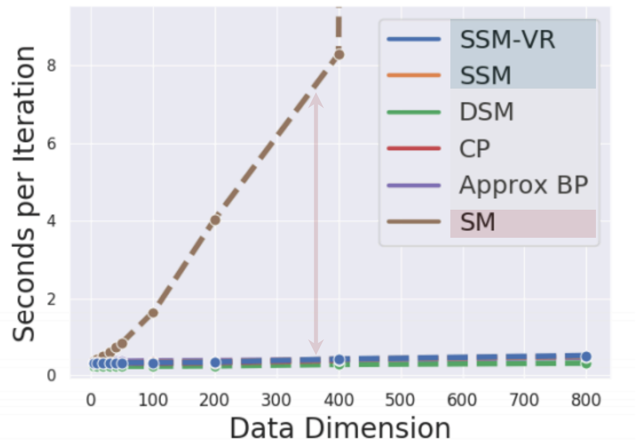


Efficiency

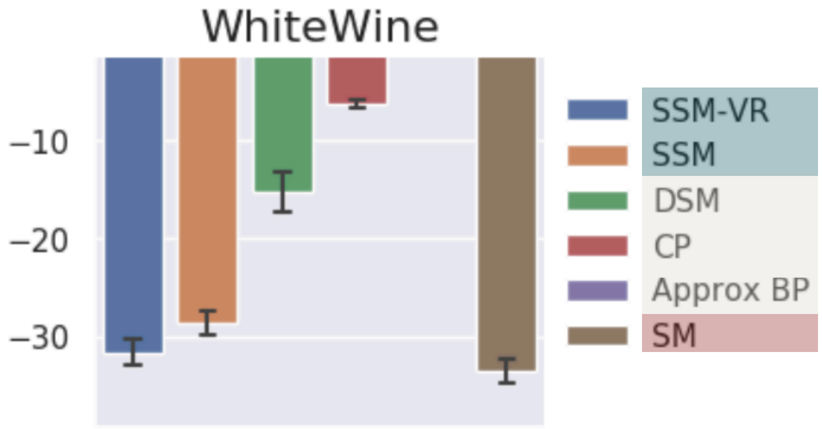
Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Results: Sliced Score Matching for EBMs

Sliced score matching methods	Other Baselines	Score Matching



Efficiency

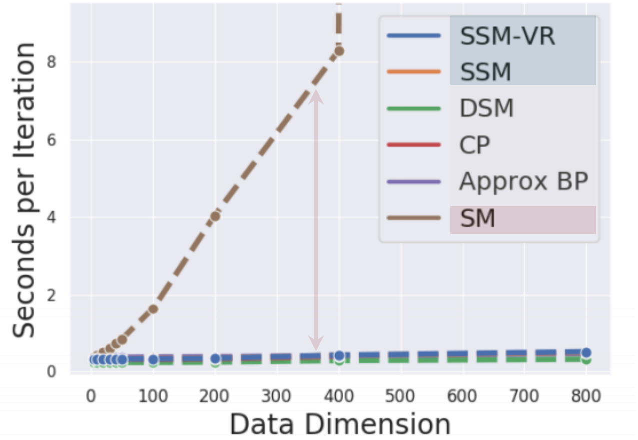


Performance on density estimation

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Results: Sliced Score Matching for EBMs

Sliced score matching methods	Other Baselines	Score Matching

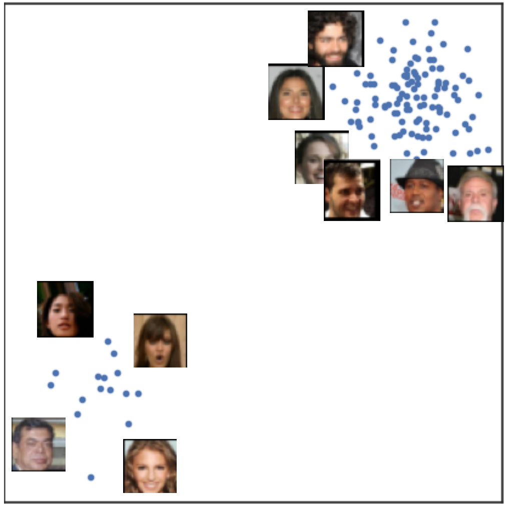


Efficiency

Performance on density estimation

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

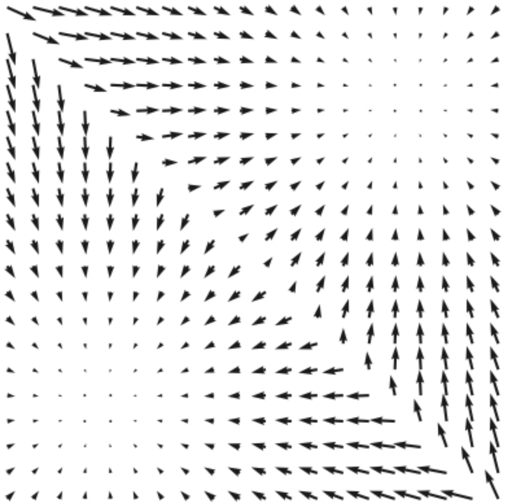
Score-based generative modeling



Data samples

$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$

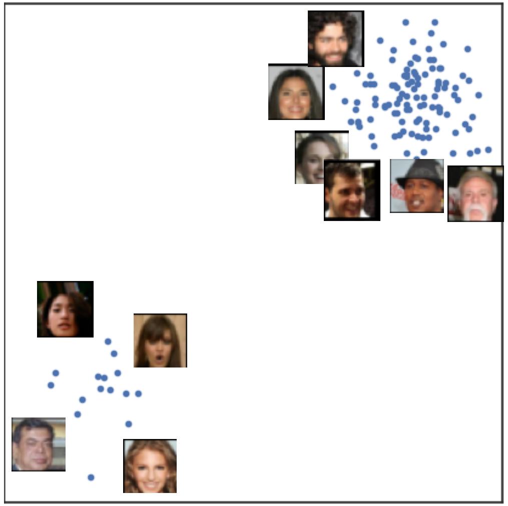
Score Matching



Scores

$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

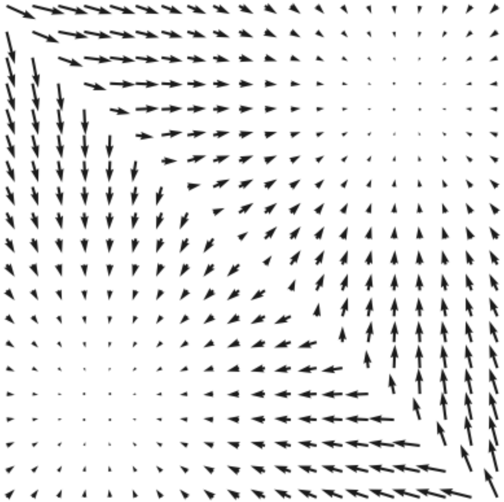
Score-based generative modeling



Data samples

$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$

Score Matching



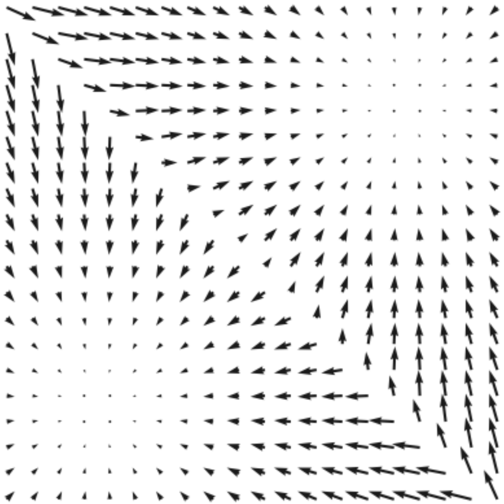
Scores

$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

?

New samples

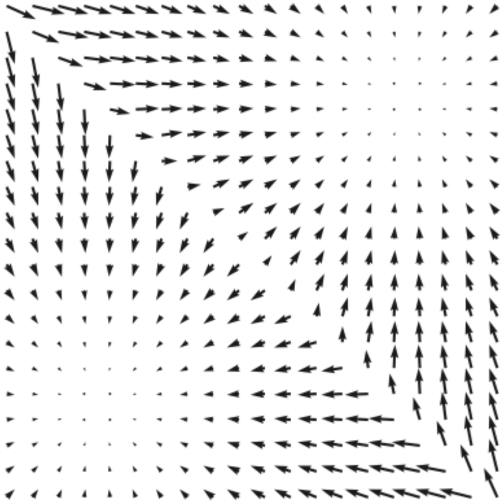
From scores to samples: Langevin MCMC



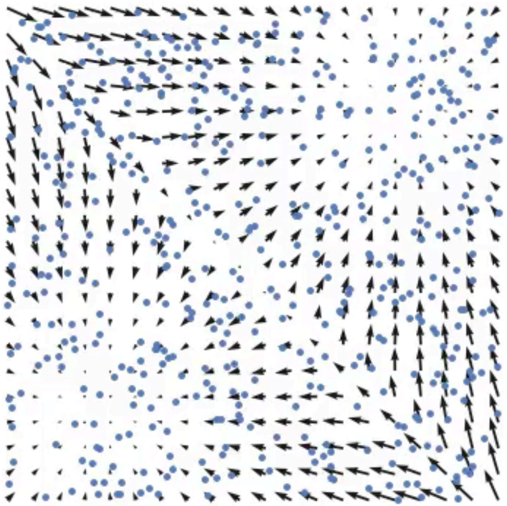
Scores

$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

From scores to samples: Langevin MCMC



Scores

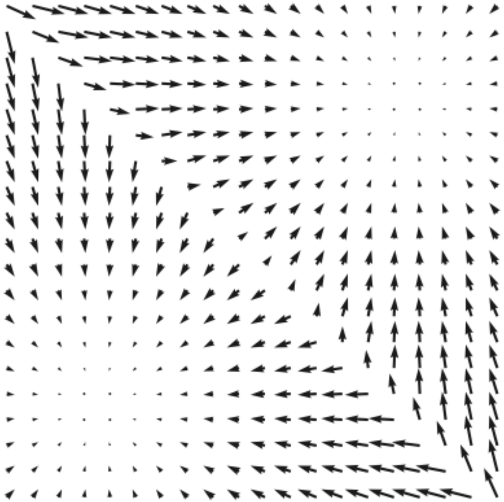


Follow the scores

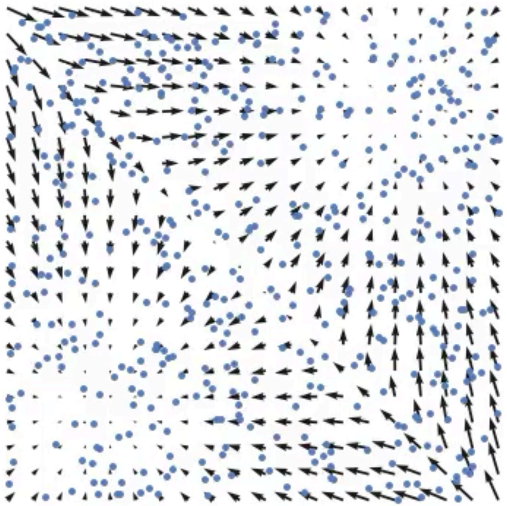
$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

$$\tilde{\mathbf{x}}_{t+1} \leftarrow \tilde{\mathbf{x}}_t + \frac{\epsilon}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t)$$

From scores to samples: Langevin MCMC



Scores

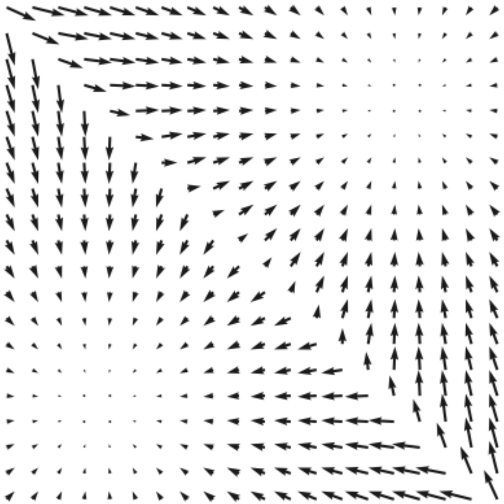


Follow the scores

$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

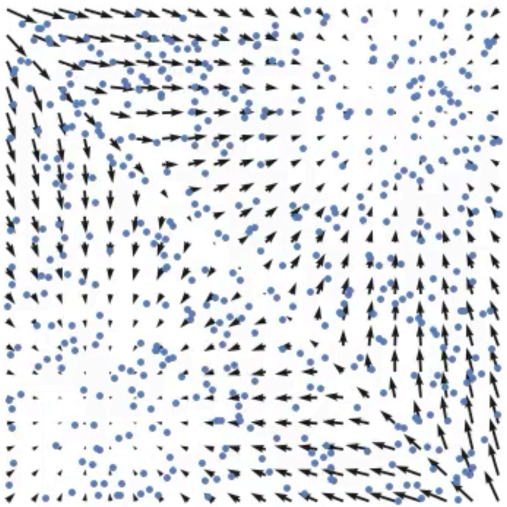
$$\tilde{\mathbf{x}}_{t+1} \leftarrow \tilde{\mathbf{x}}_t + \frac{\epsilon}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t)$$

From scores to samples: Langevin MCMC



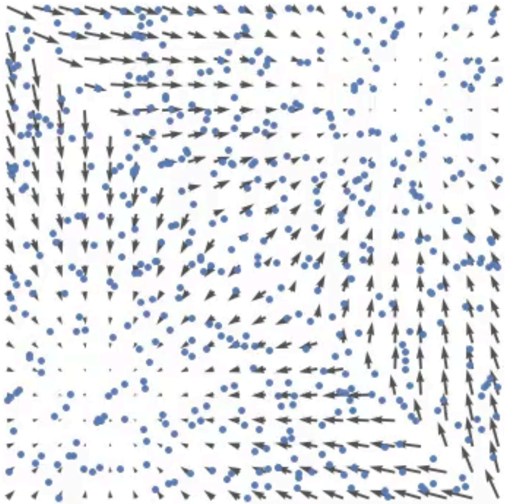
Scores

$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$



Follow the scores

$$\tilde{\mathbf{x}}_{t+1} \leftarrow \tilde{\mathbf{x}}_t + \frac{\epsilon}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t)$$

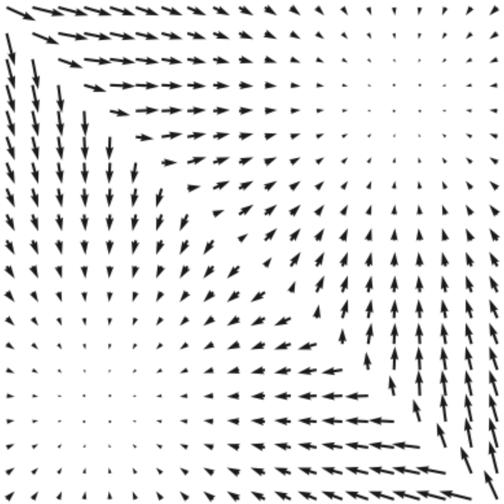


Follow noisy scores:
Langevin MCMC

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

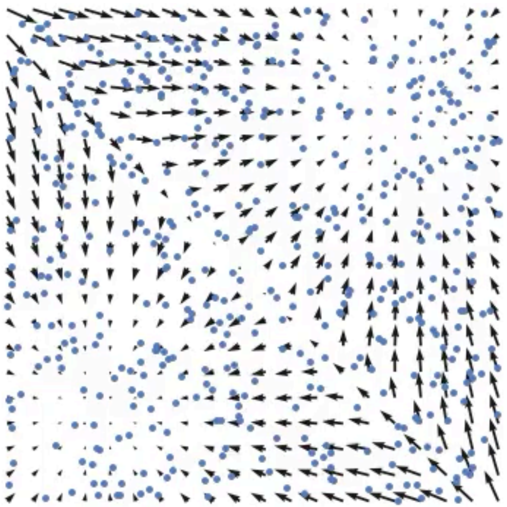
$$\tilde{\mathbf{x}}_{t+1} \leftarrow \tilde{\mathbf{x}}_t + \frac{\epsilon}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t) + \sqrt{\epsilon}\,\mathbf{z}_t$$

From scores to samples: Langevin MCMC



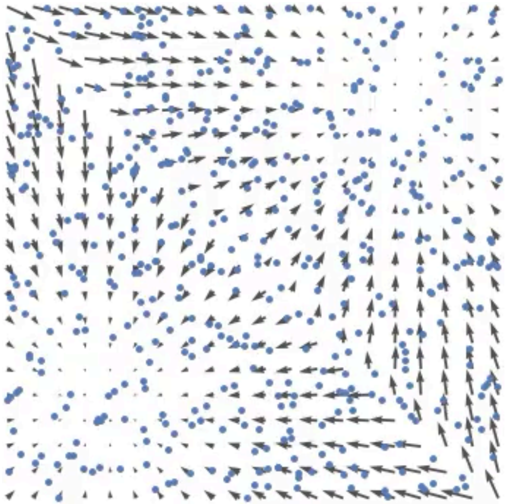
Scores

$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$



Follow the scores

$$\tilde{\mathbf{x}}_{t+1} \leftarrow \tilde{\mathbf{x}}_t + \frac{\epsilon}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t)$$

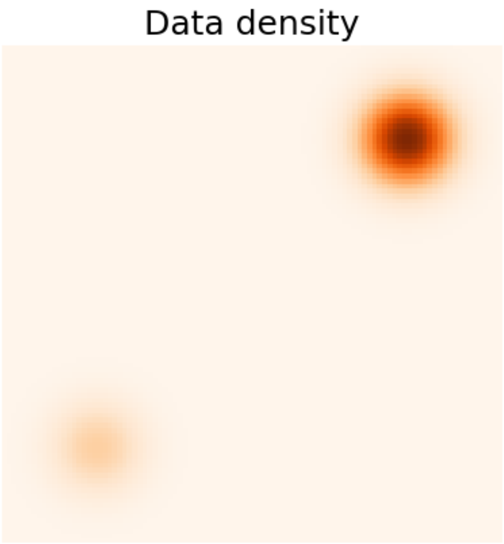


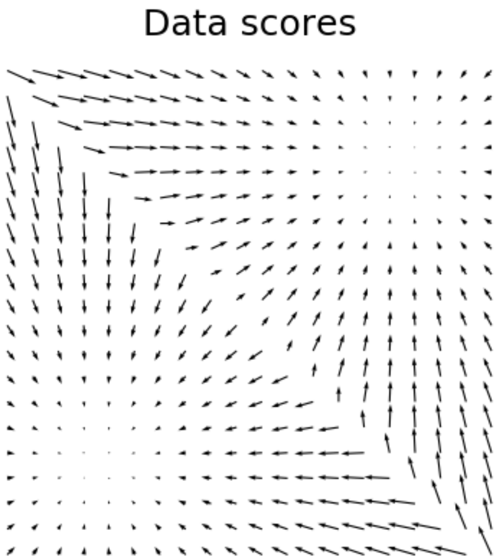
Follow noisy scores: Langevin MCMC

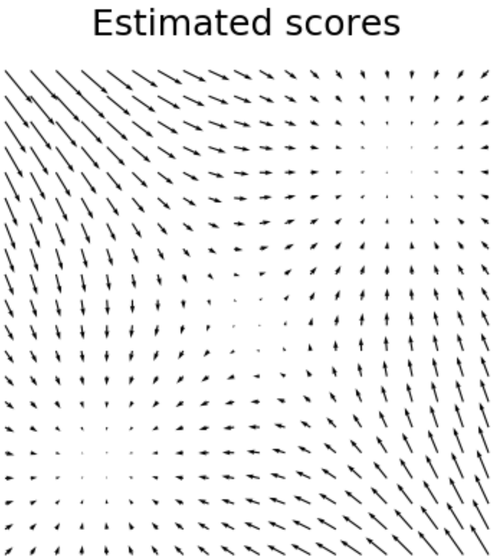
$$\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

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

Challenge in low data density regions

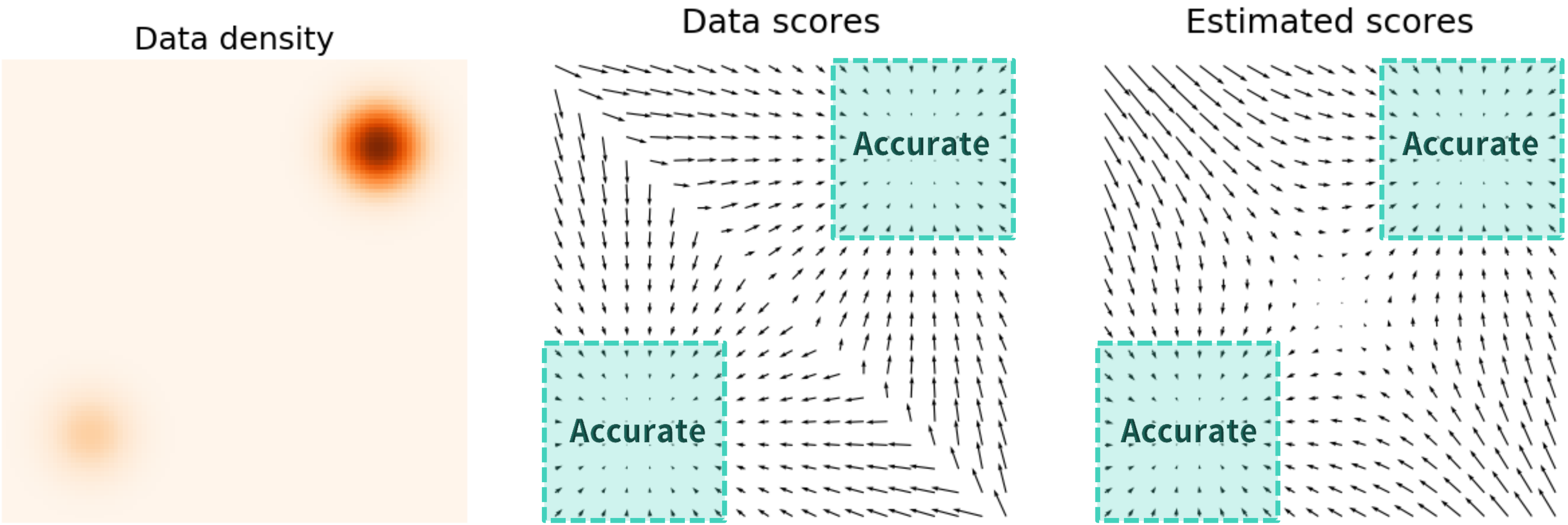






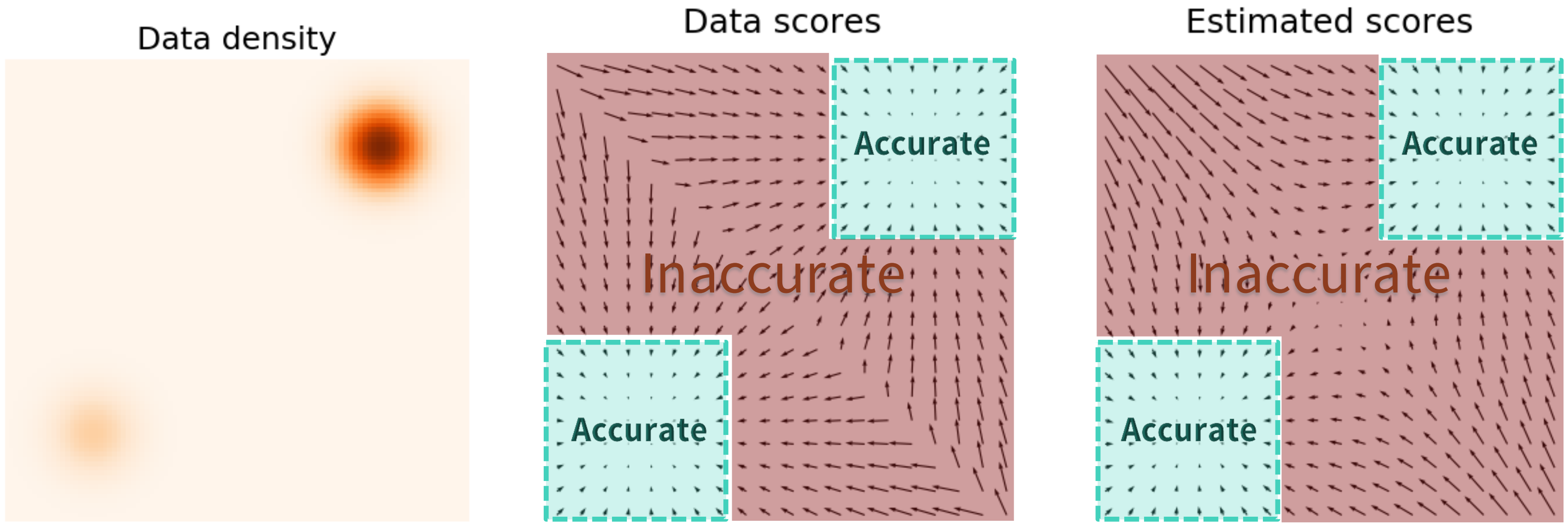
Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

Challenge in low data density regions



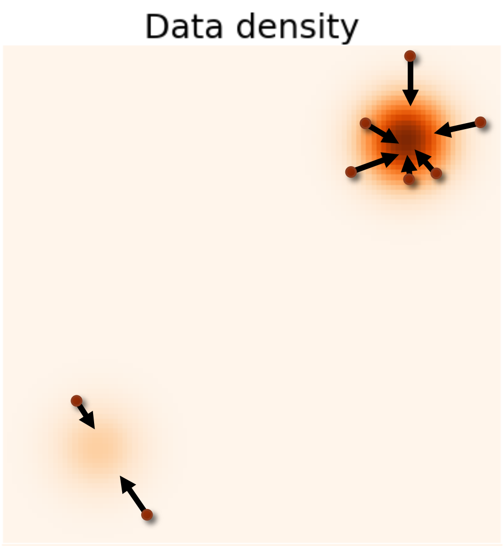
Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Challenge in low data density regions

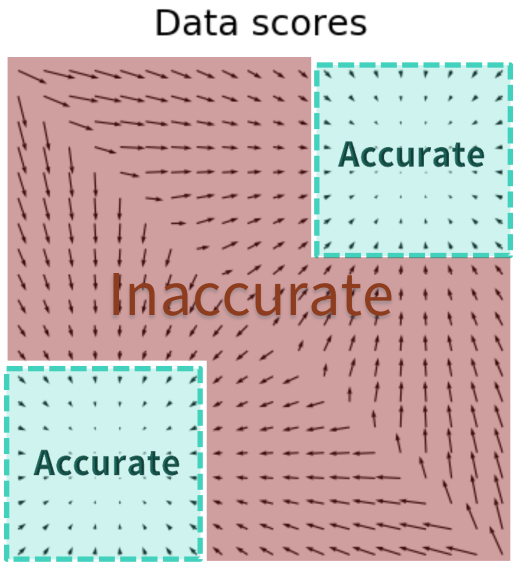


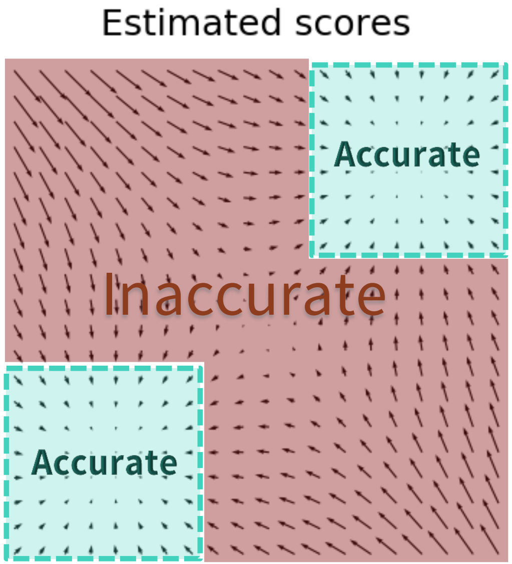
Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

Challenge in low data density regions



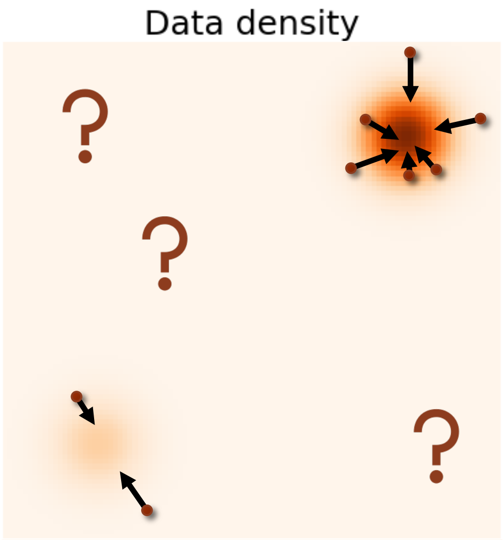
$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$



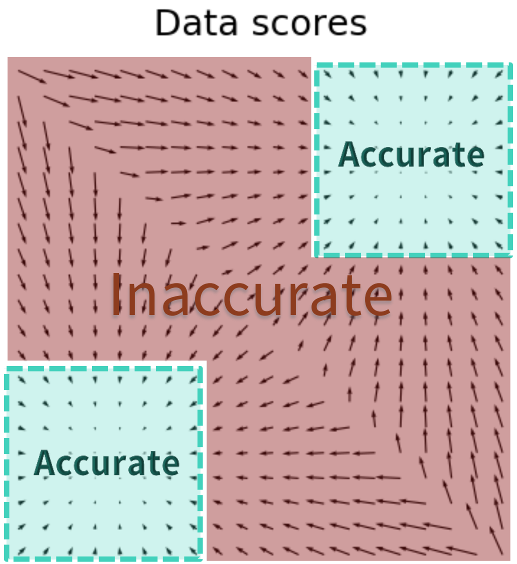


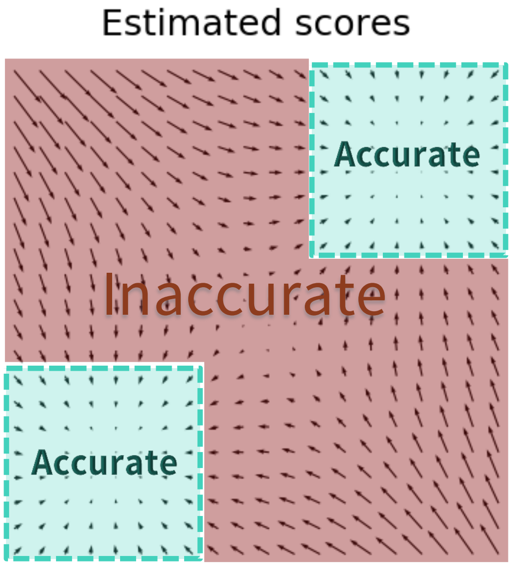
Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

Challenge in low data density regions



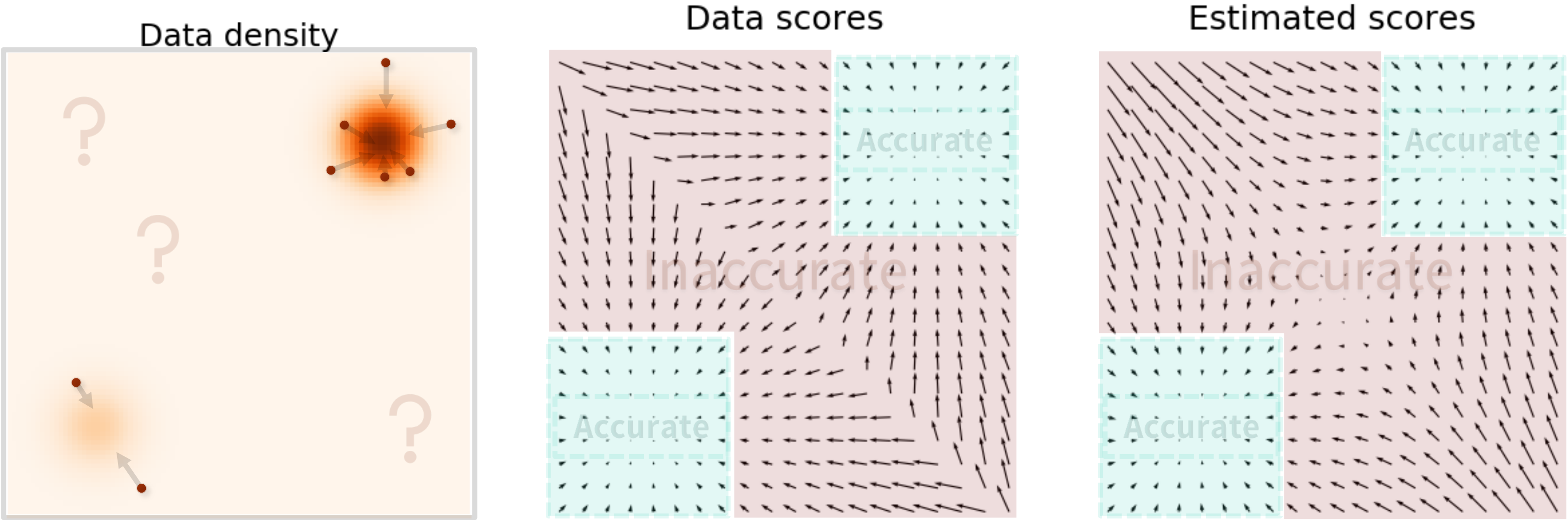
$$\frac{1}{2}\mathbb{E}_{p_{\mathrm{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\mathrm{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$





Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

Challenge in low data density regions

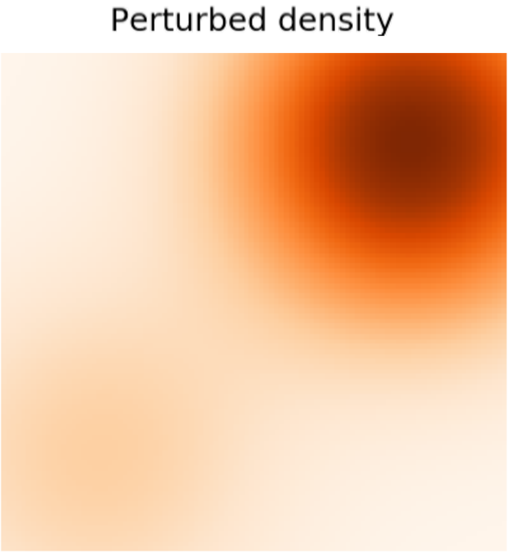


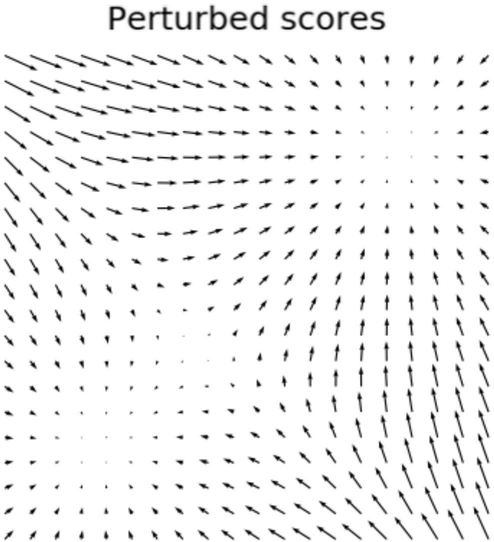
$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$

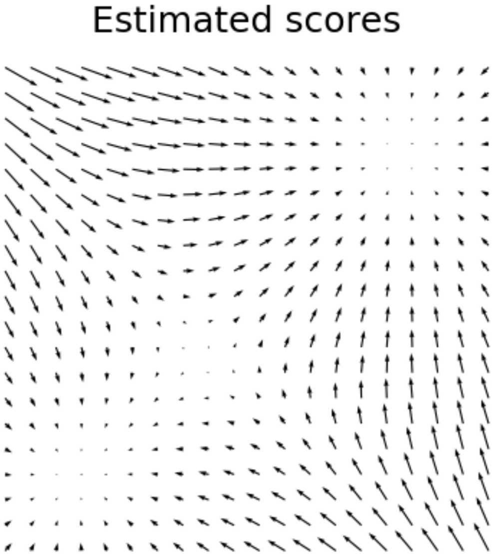
Langevin MCMC will have trouble exploring low density regions

Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

Adding noise to data

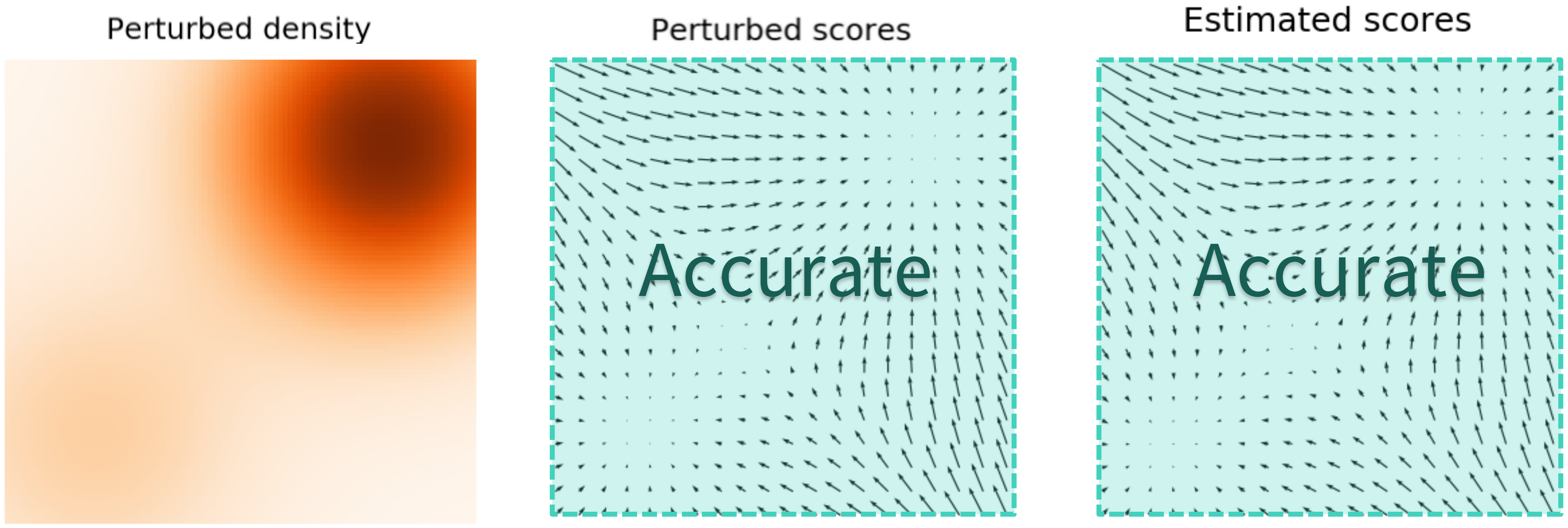






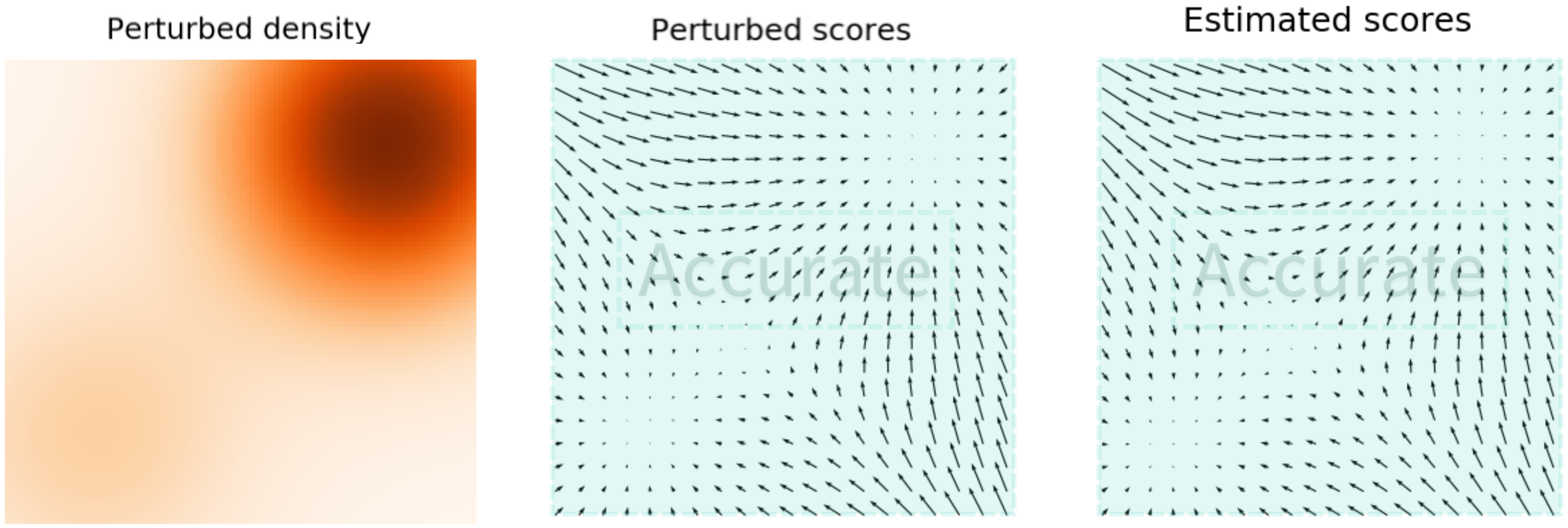
Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Adding noise to data



Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

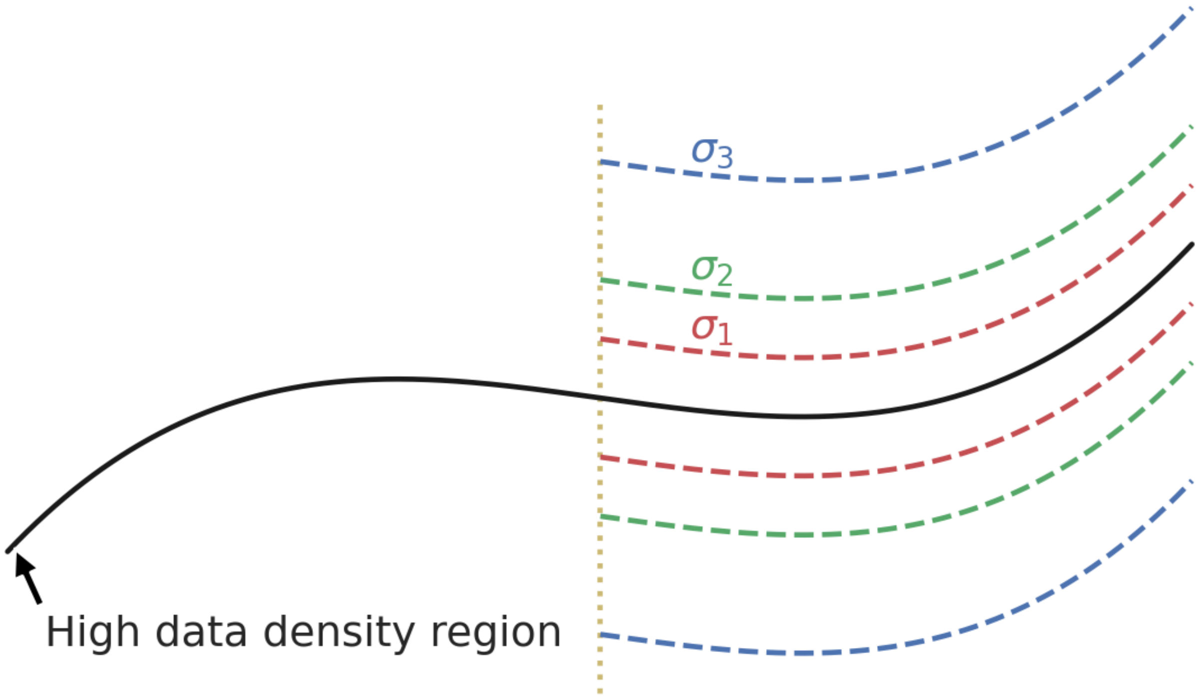
Adding noise to data



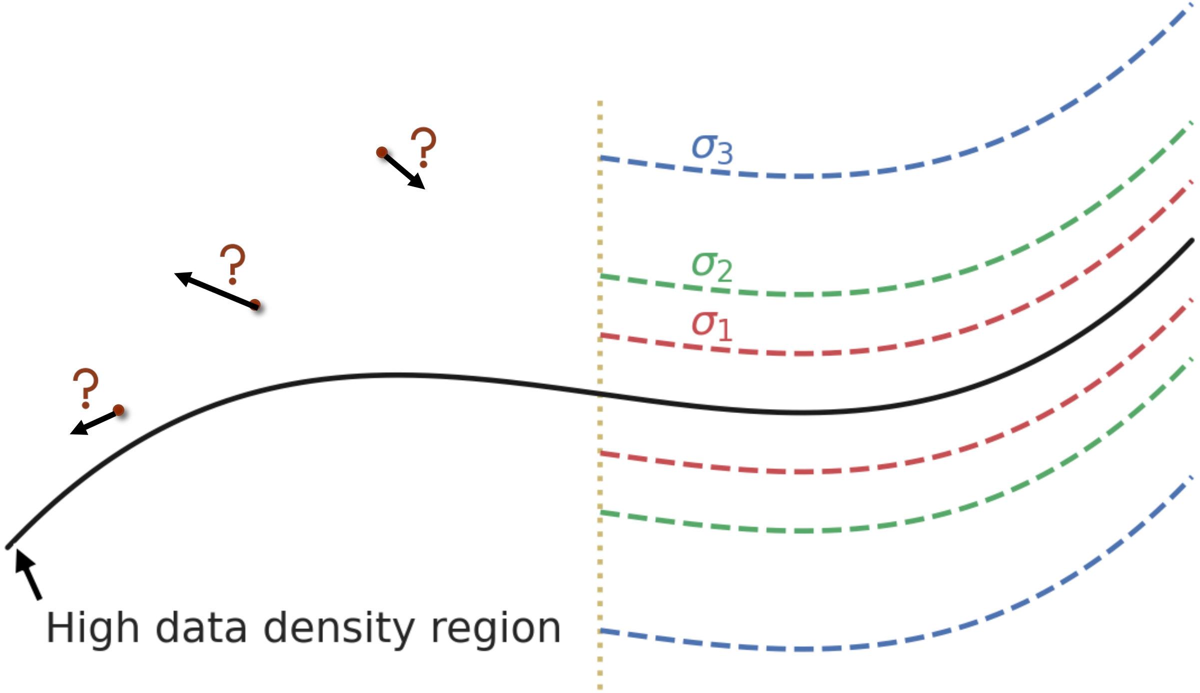
Provide useful directional information for Langevin MCMC.

Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

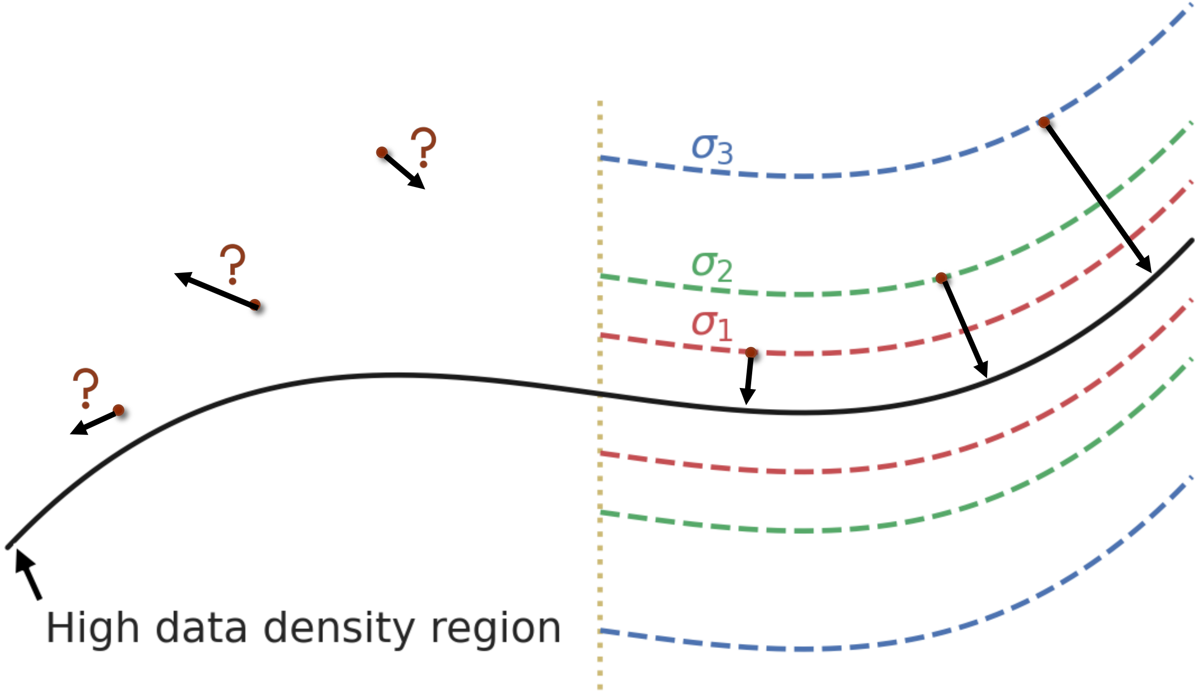
Using multiple noise scales



Using multiple noise scales



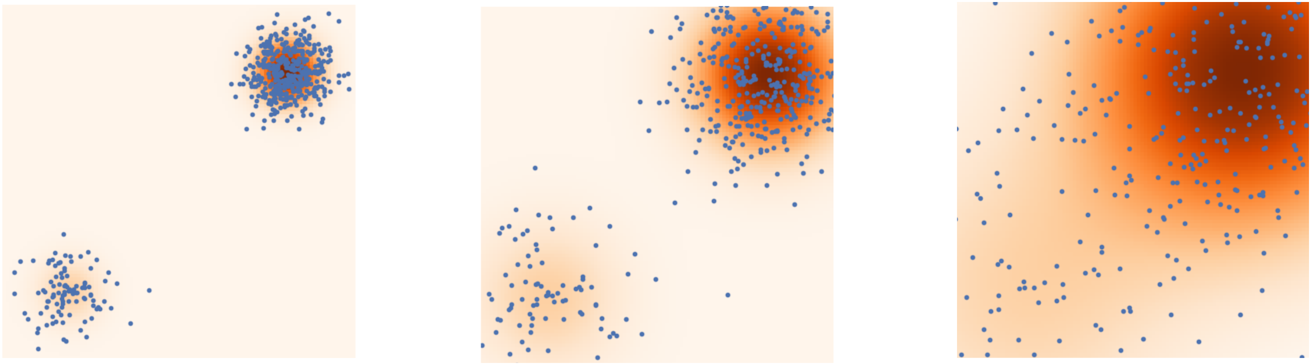
Using multiple noise scales



Using multiple noise scales

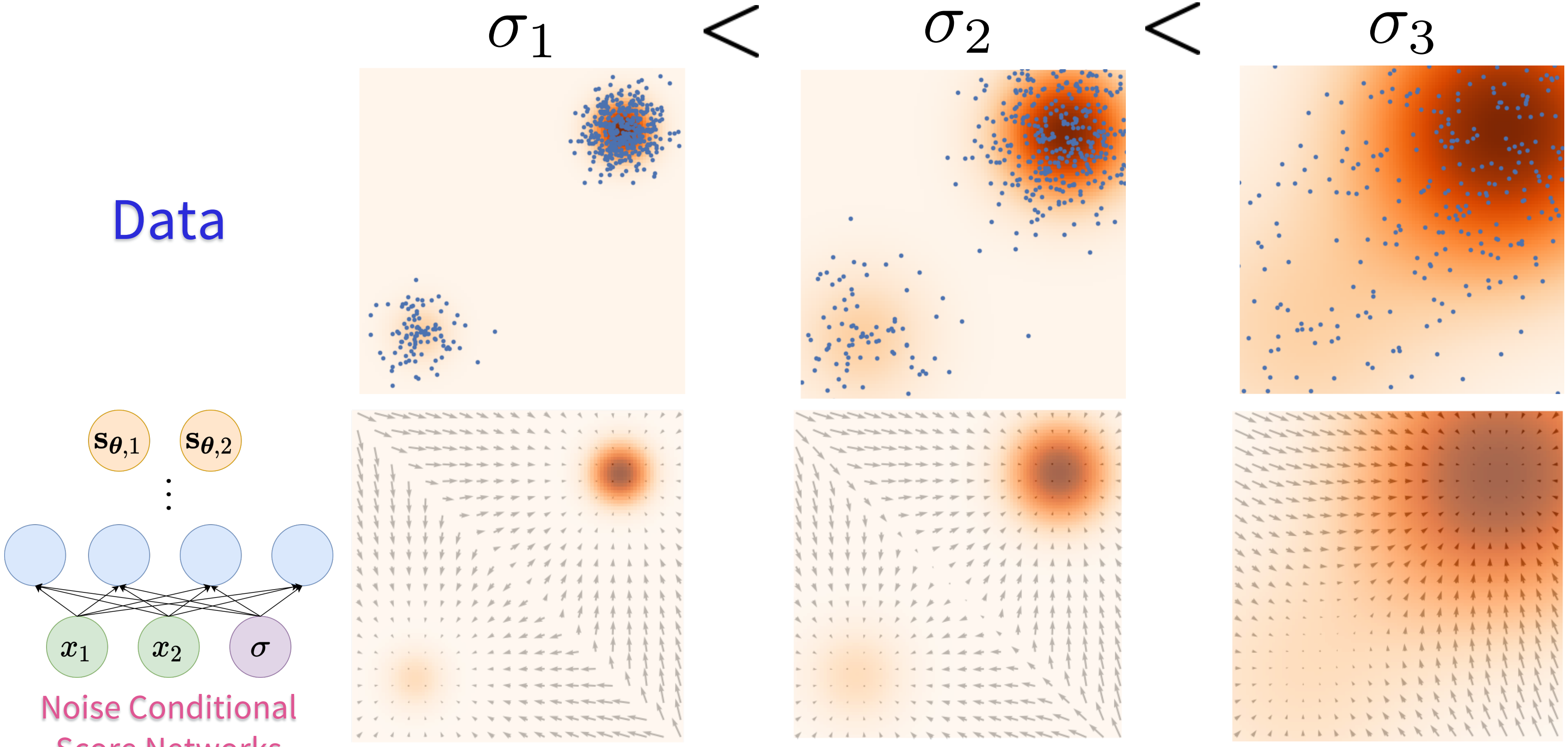
$\sigma_1 < \sigma_2 < \sigma_3$

Data



Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales



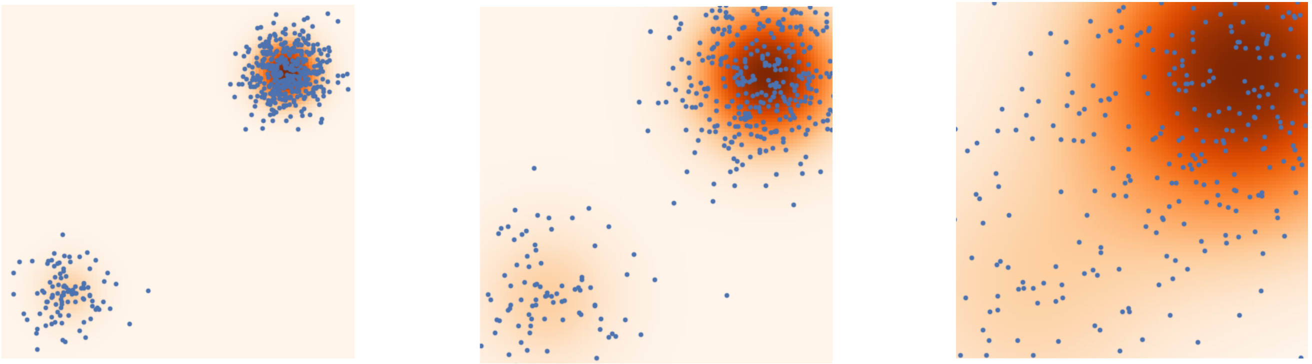
Noise Conditional Score Networks (NCSN)

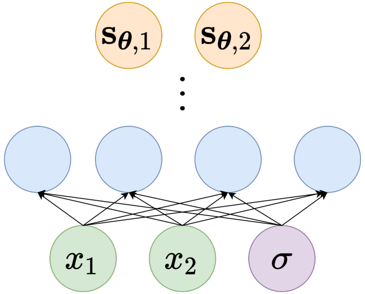
Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales

$\sigma_1 < \sigma_2 < \sigma_3$

Data





Noise Conditional Score Networks (NCSN)

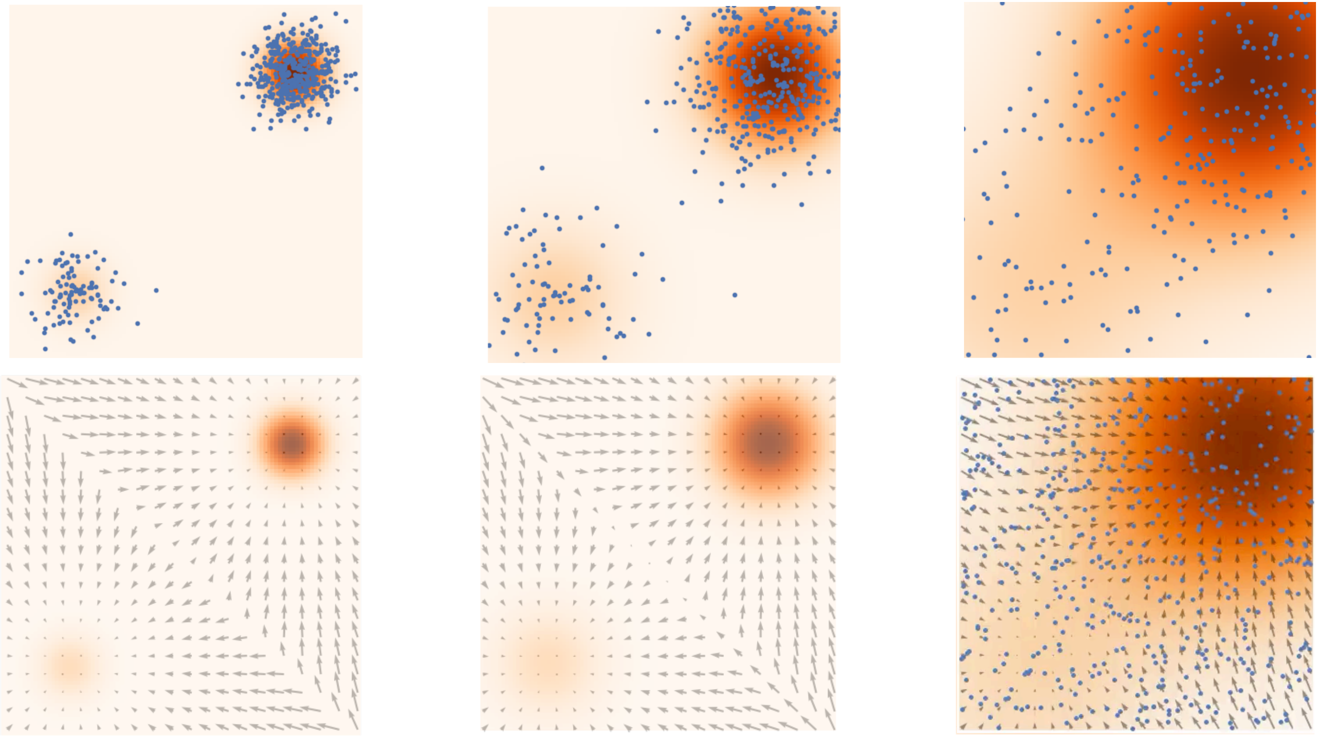
$$\sum_{i=1}^{N} \lambda(\sigma_i)\mathbb{E}_{p_{\sigma_i}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, \sigma_i)\|_2^2]$$

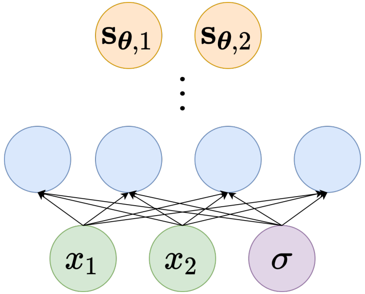
Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales

$$\sigma_1 < \sigma_2 < \sigma_3$$

Data

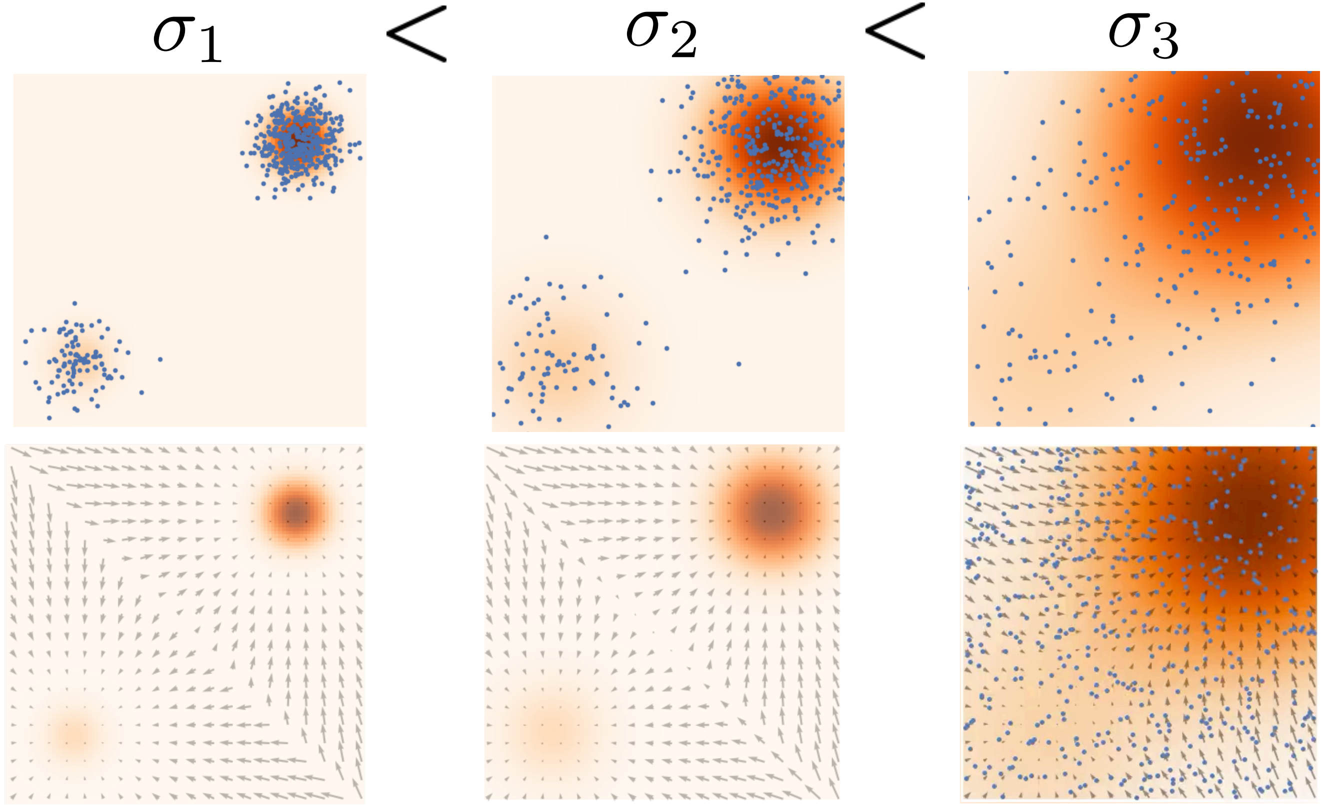


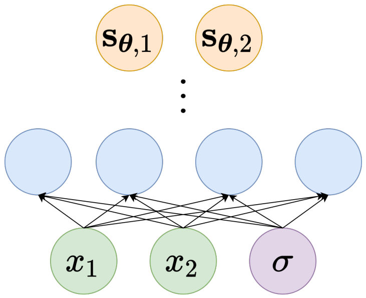


Noise Conditional Score Networks (NCSN)

Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

Using multiple noise scales

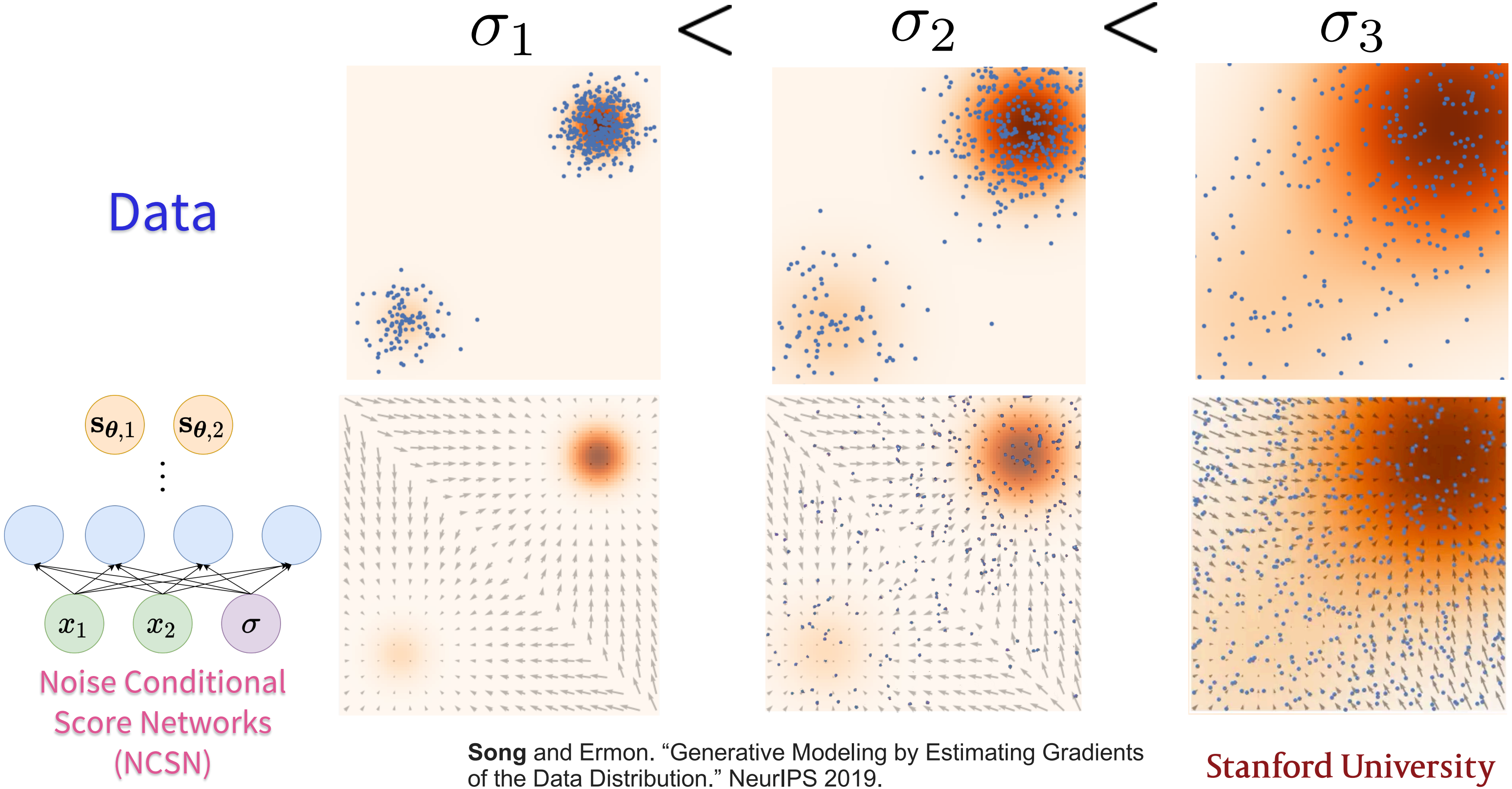




Noise Conditional Score Networks (NCSN)

Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales

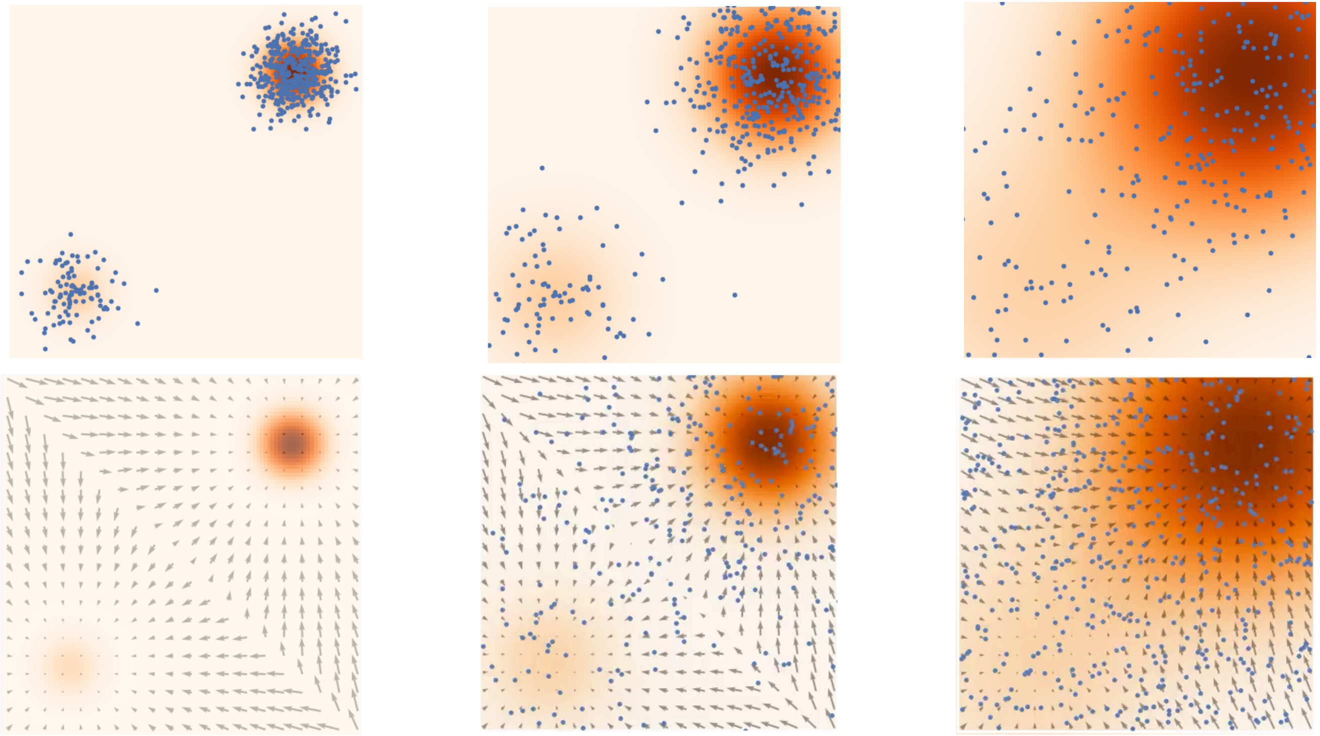


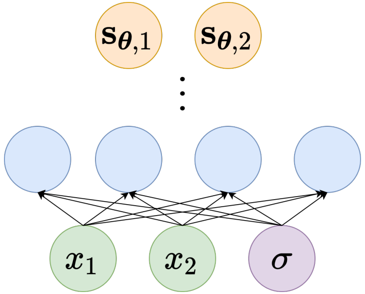
Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales

$$\sigma_1 < \sigma_2 < \sigma_3$$

Data

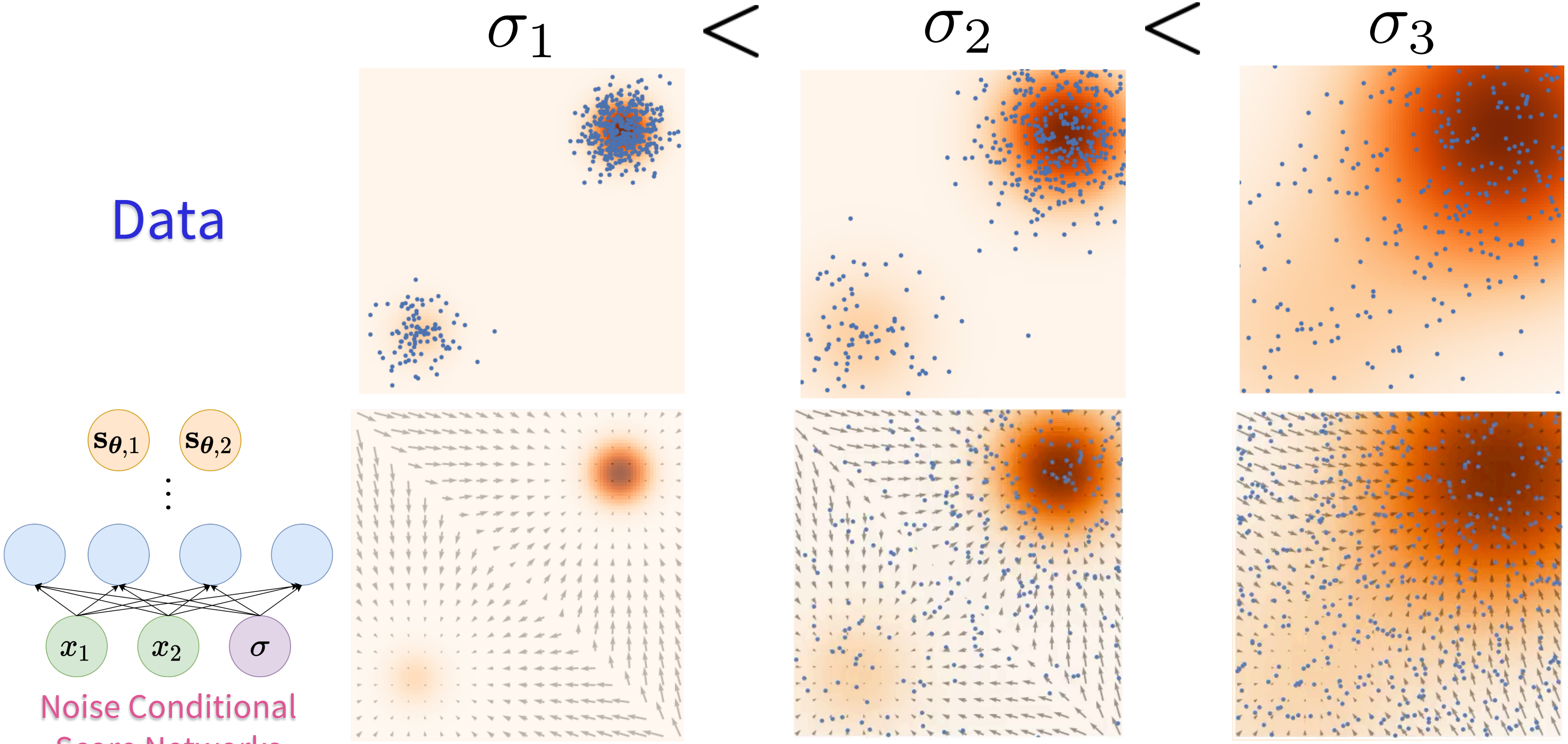




Noise Conditional Score Networks (NCSN)

Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

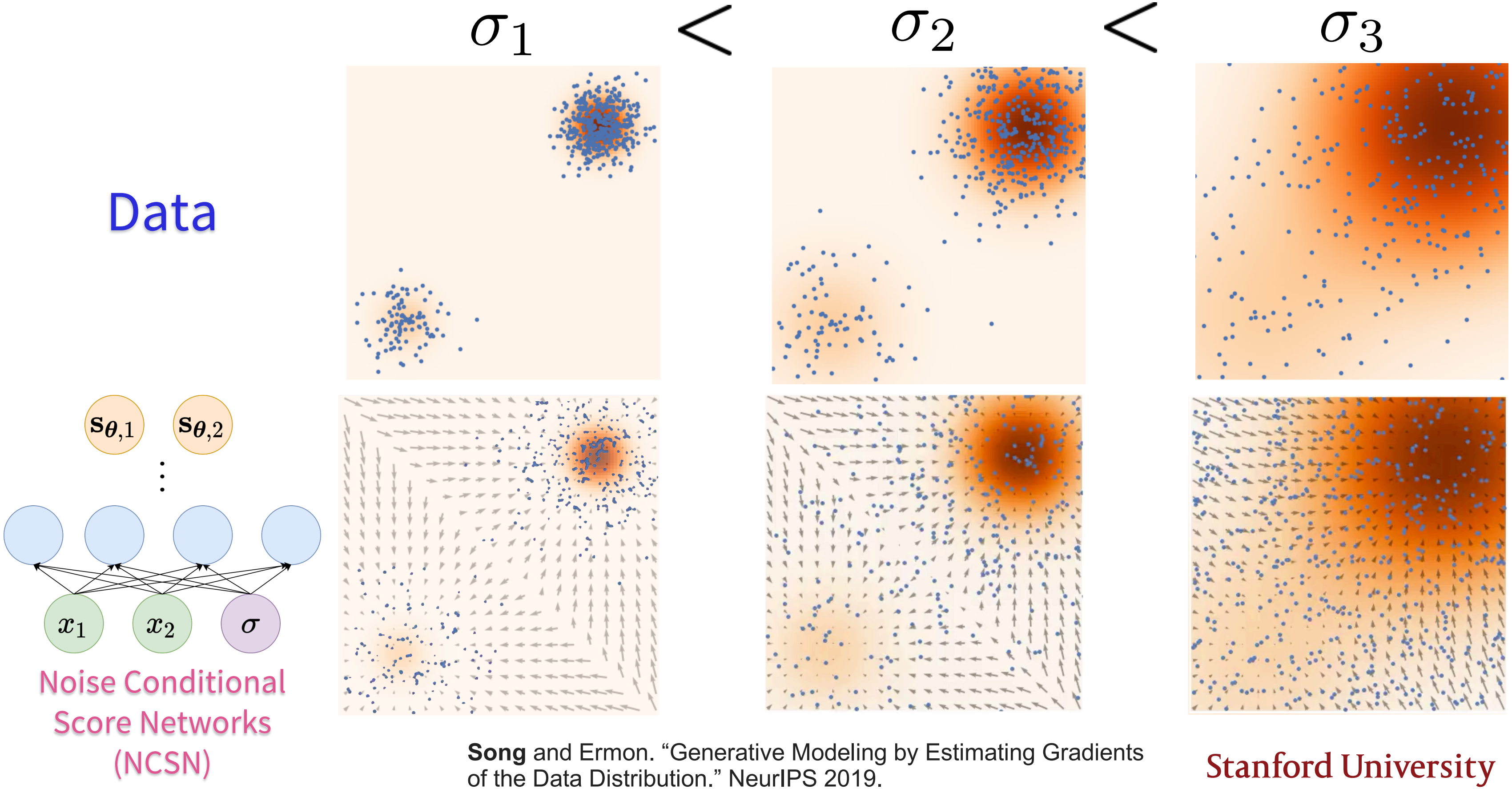
Using multiple noise scales



Noise Conditional Score Networks (NCSN)

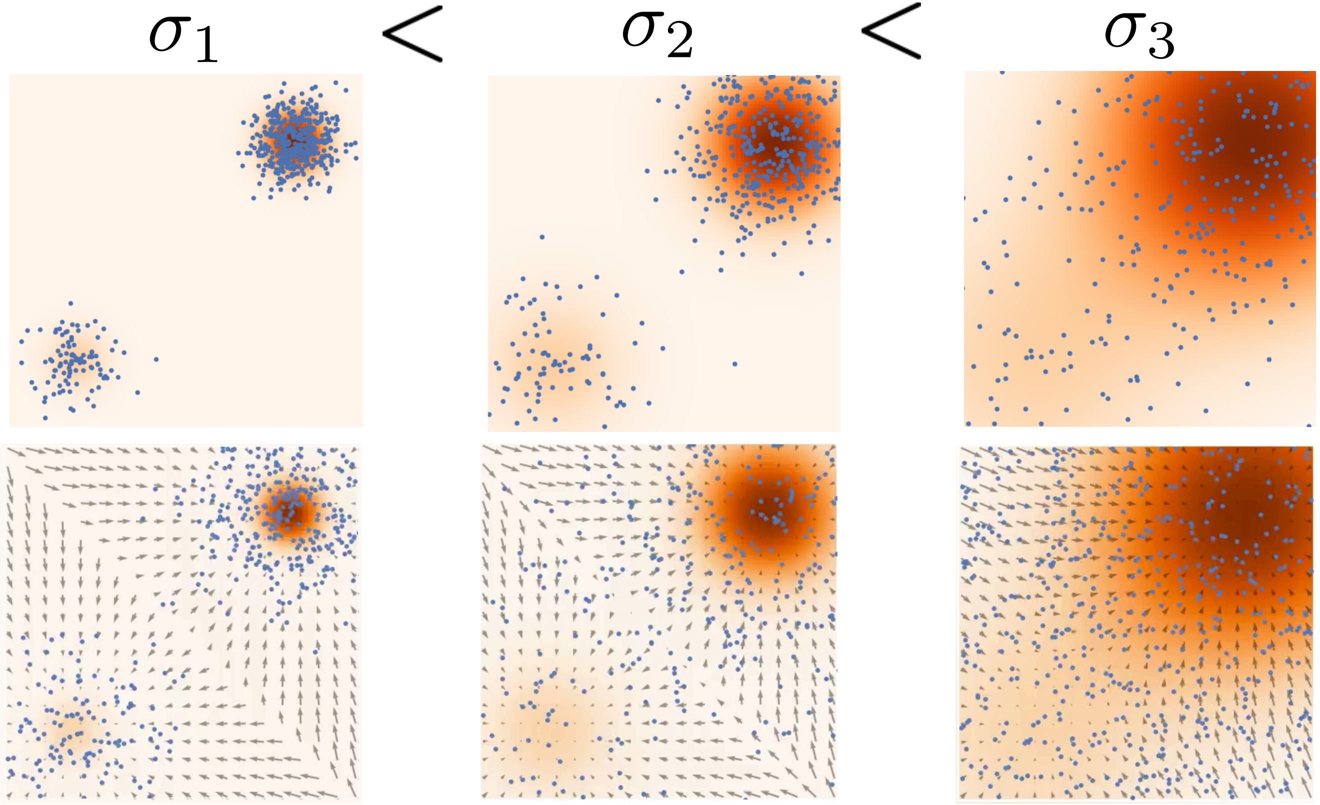
Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales

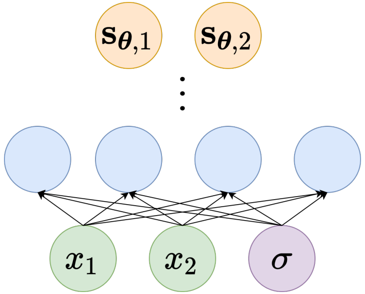


Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales



Data



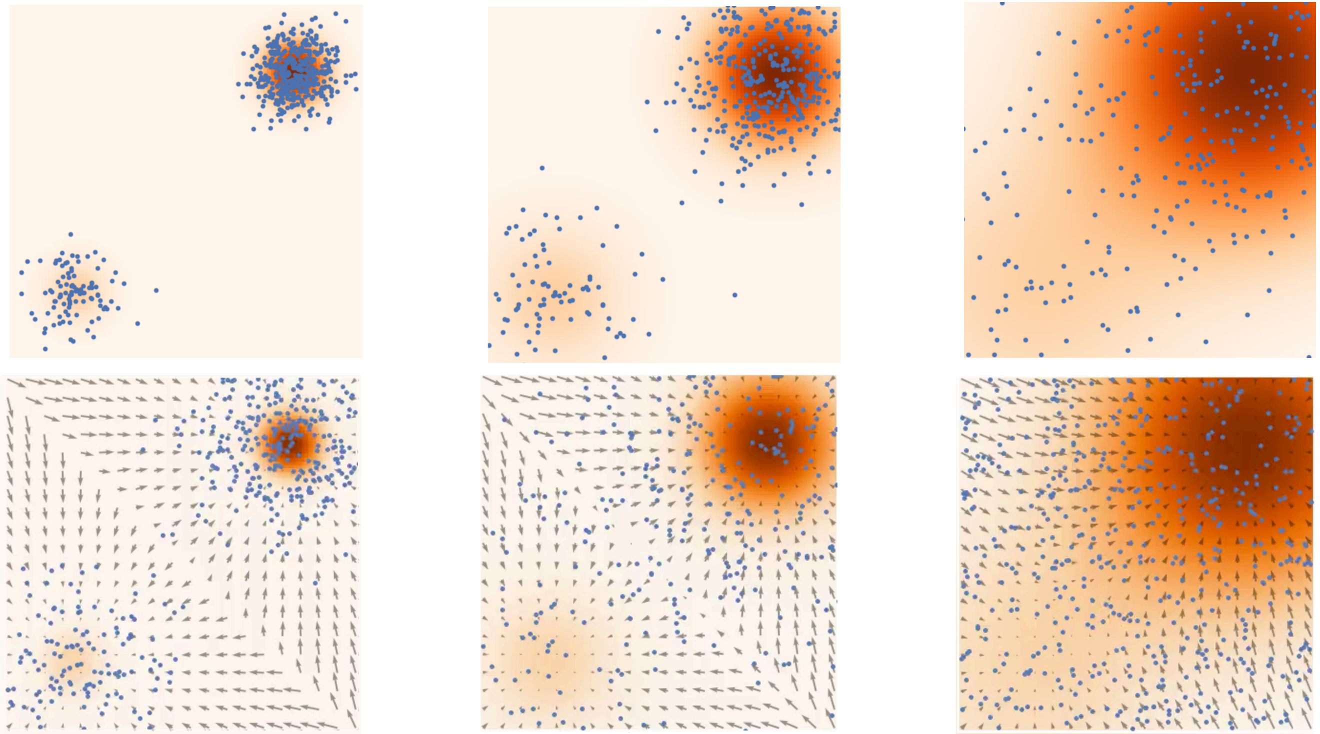
Noise Conditional Score Networks (NCSN)

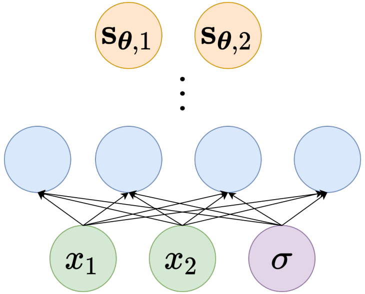
Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales

$\sigma_1 < \sigma_2 < \sigma_3$

Data



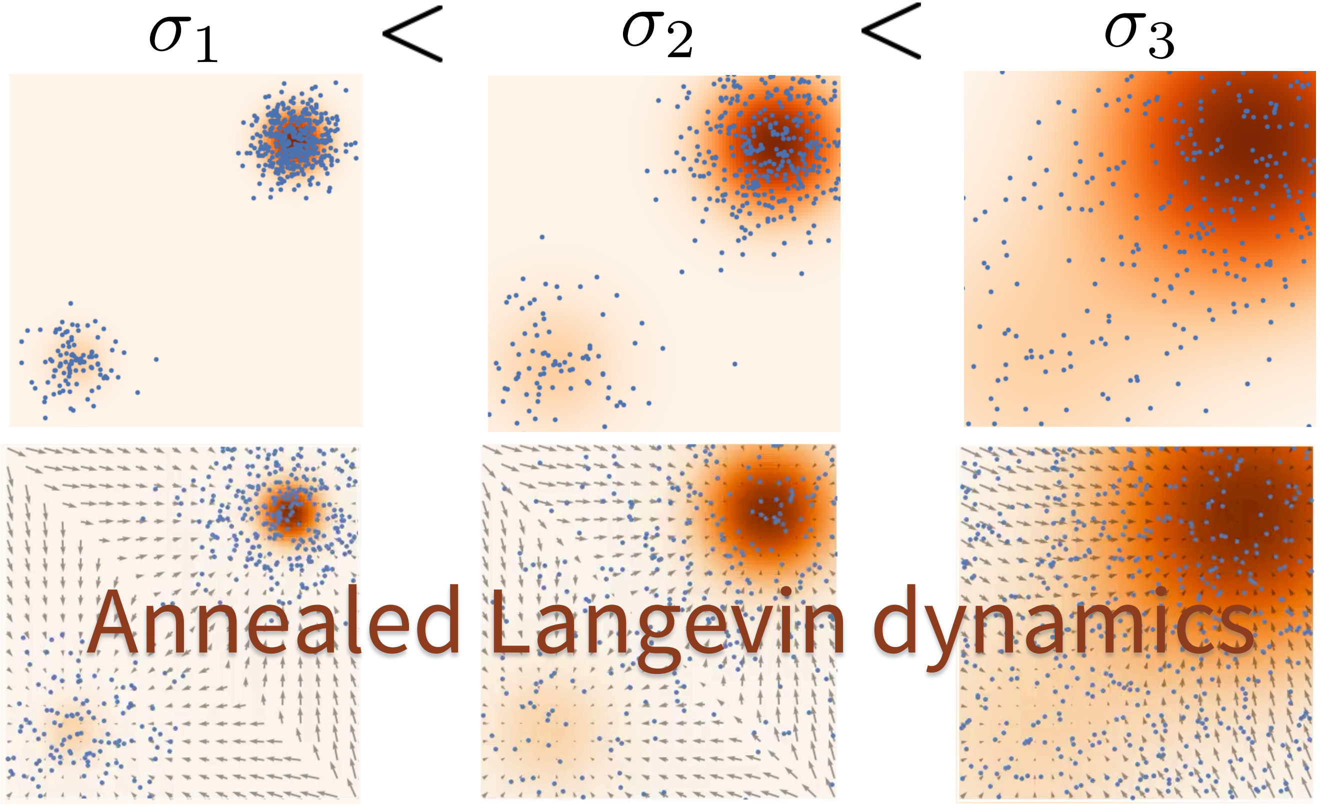


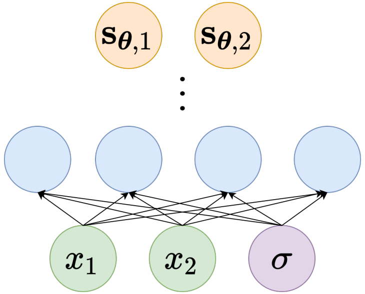
Noise Conditional Score Networks (NCSN)

Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Using multiple noise scales

Data

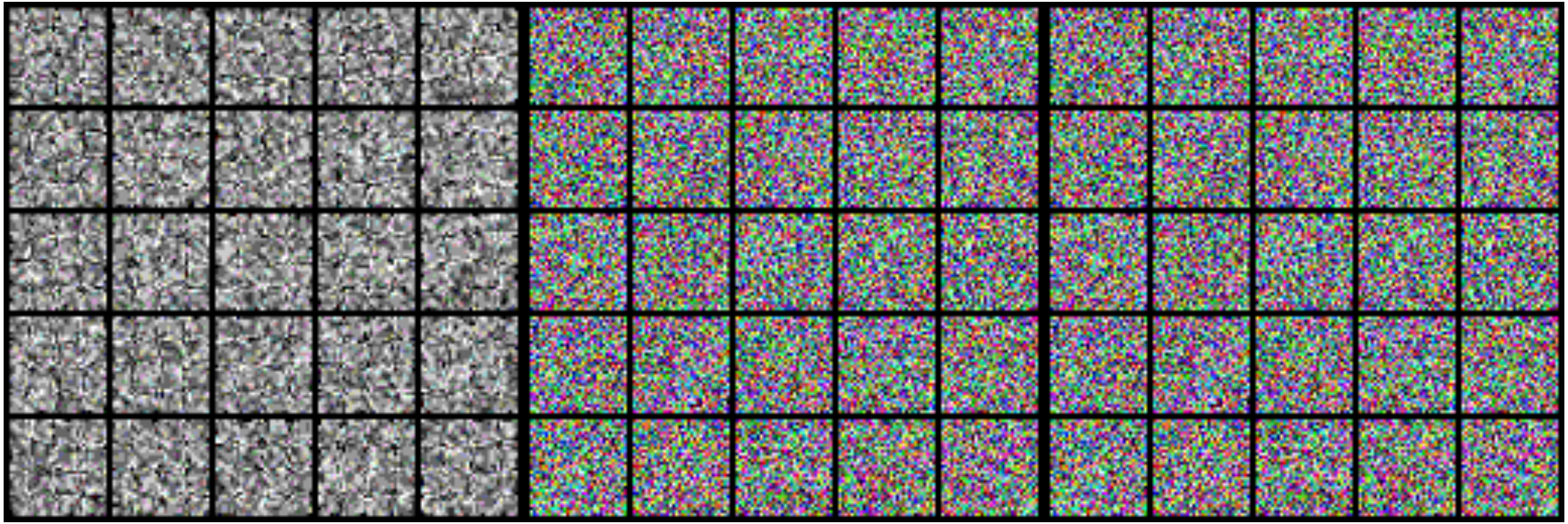




Noise Conditional Score Networks (NCSN)

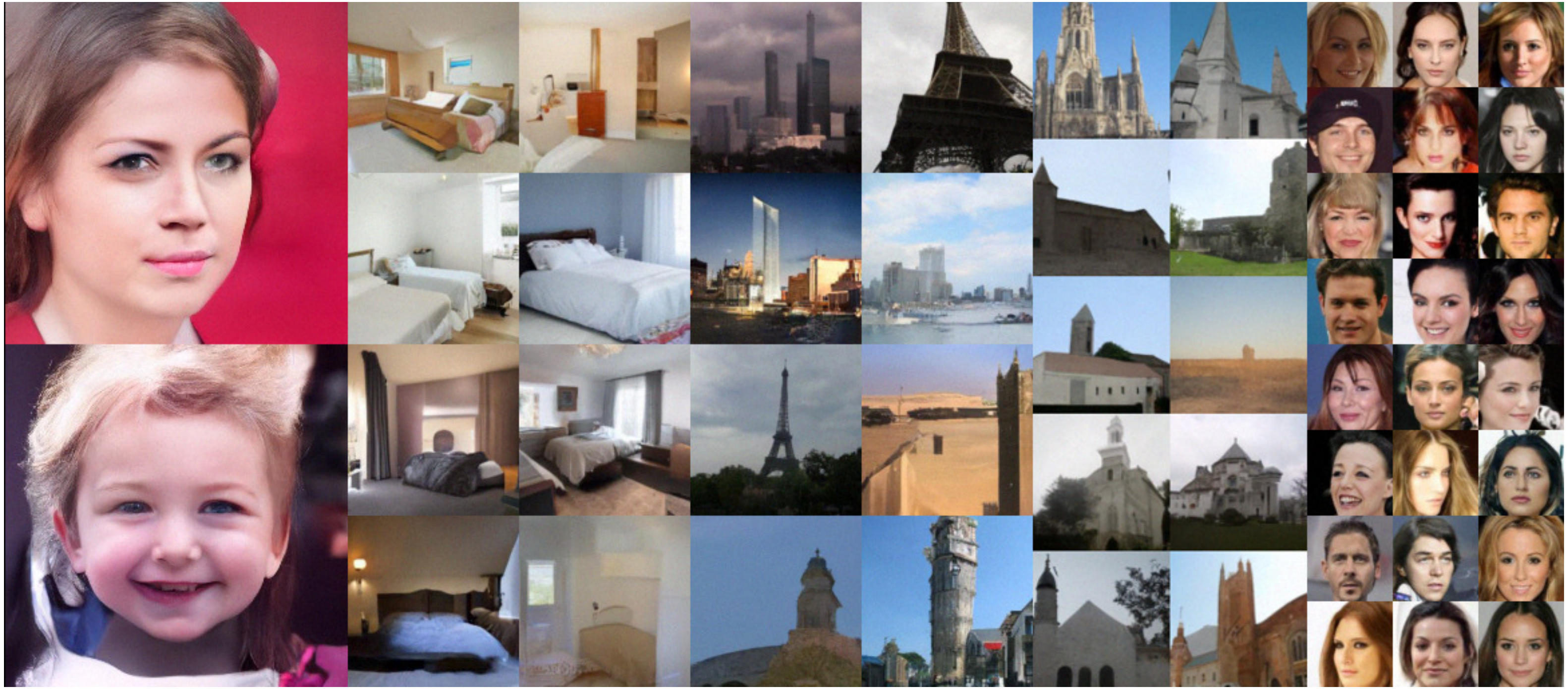
Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

Sampling in the real world



Song and Ermon. “Generative Modeling by Estimating Gradients of the Data Distribution.” NeurIPS 2019.

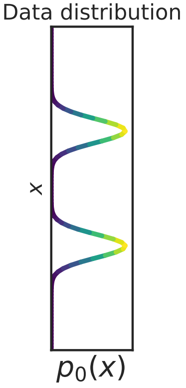
High resolution image generation



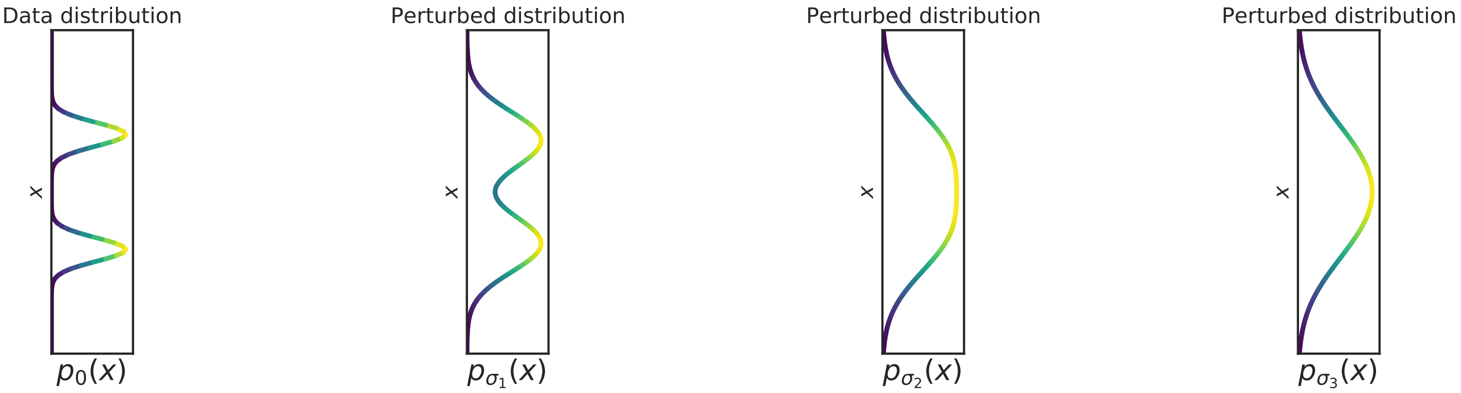
Song and Ermon. “Improved Techniques for Training Score-Based Generative Modles.” NeurIPS 2020.

Using an infinite number of noise scales

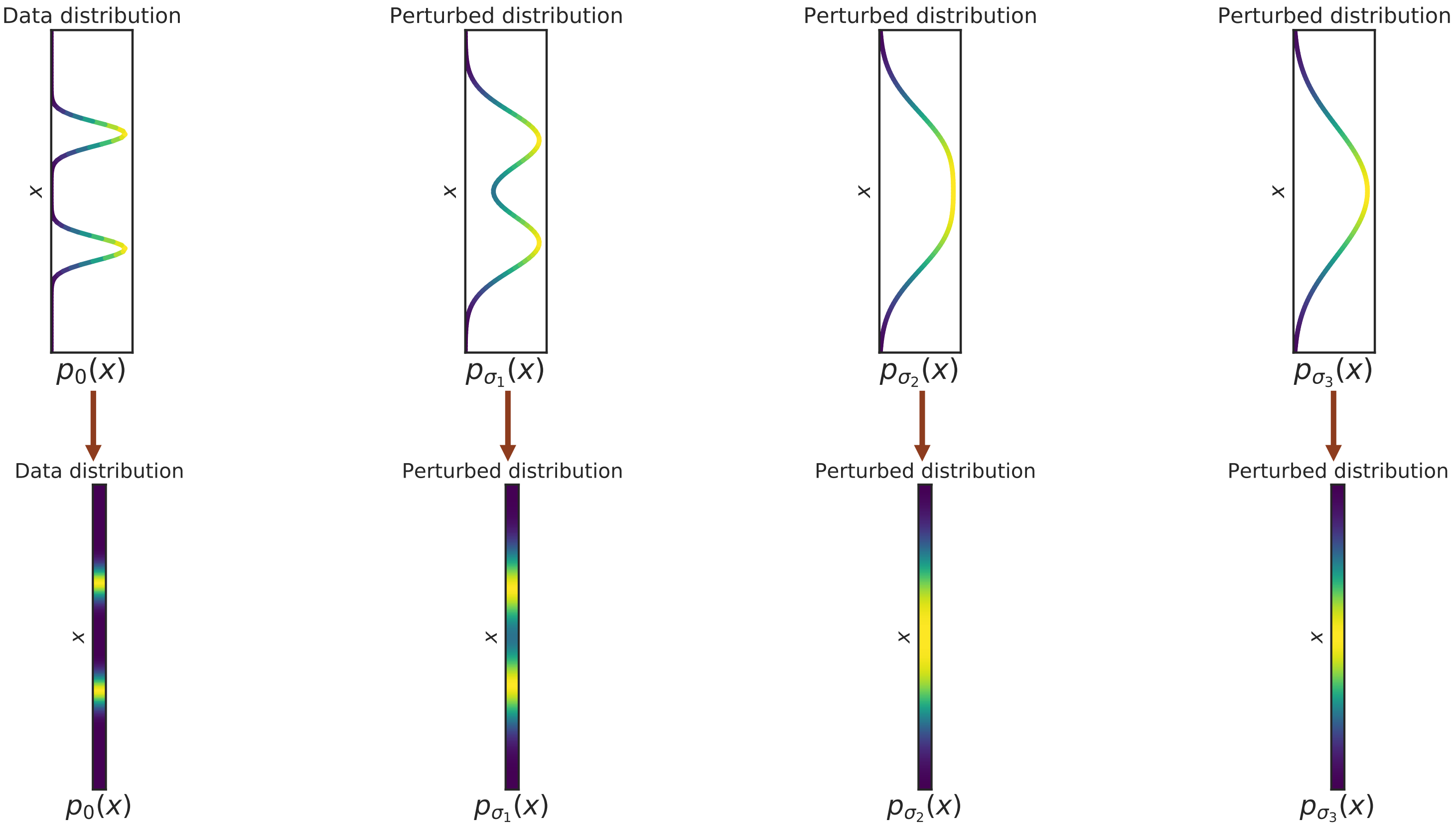
Using an infinite number of noise scales



Using an infinite number of noise scales

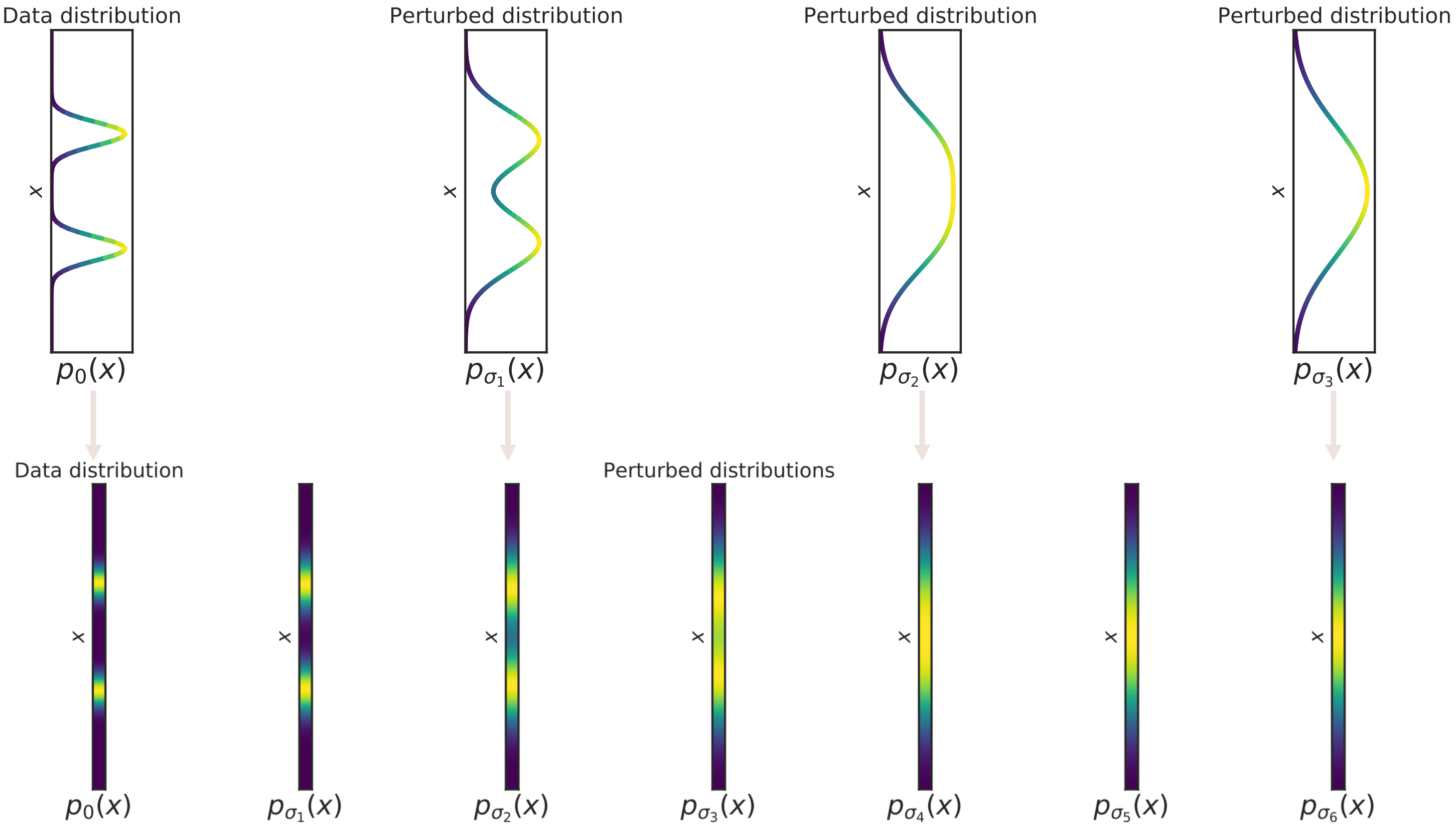


Using an infinite number of noise scales



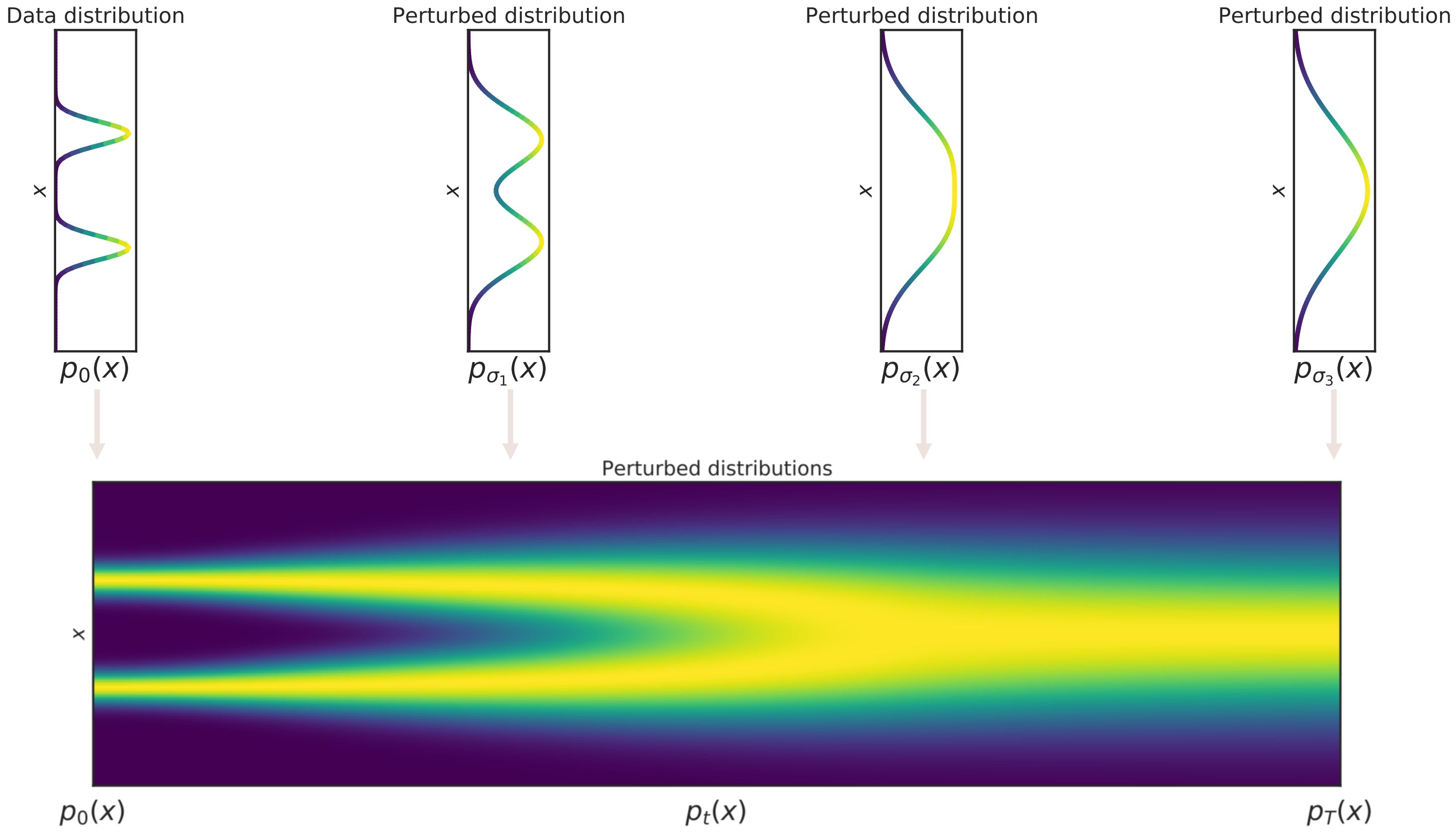
Stanford University

Using an infinite number of noise scales

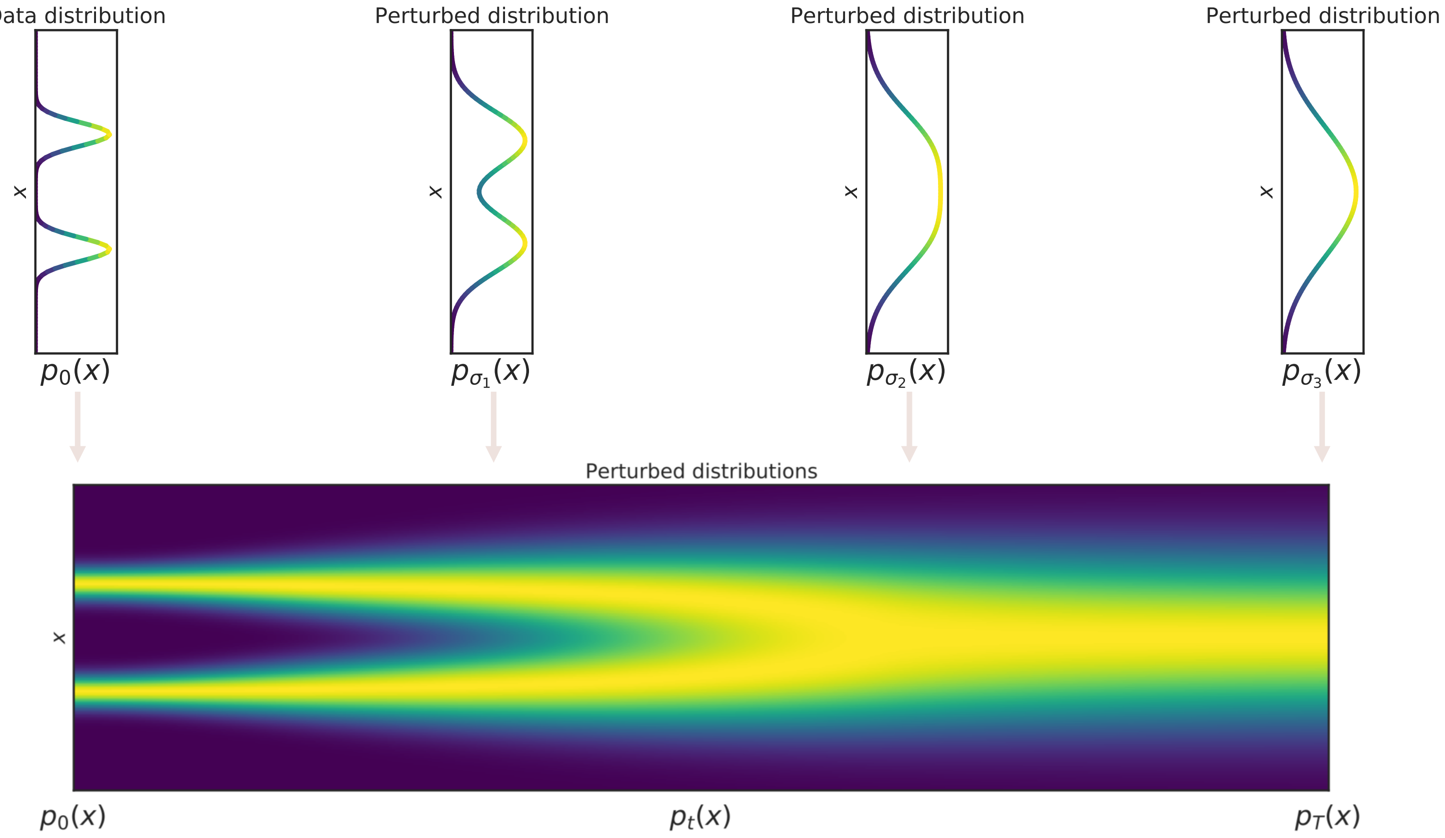


Stanford University

Using an infinite number of noise scales

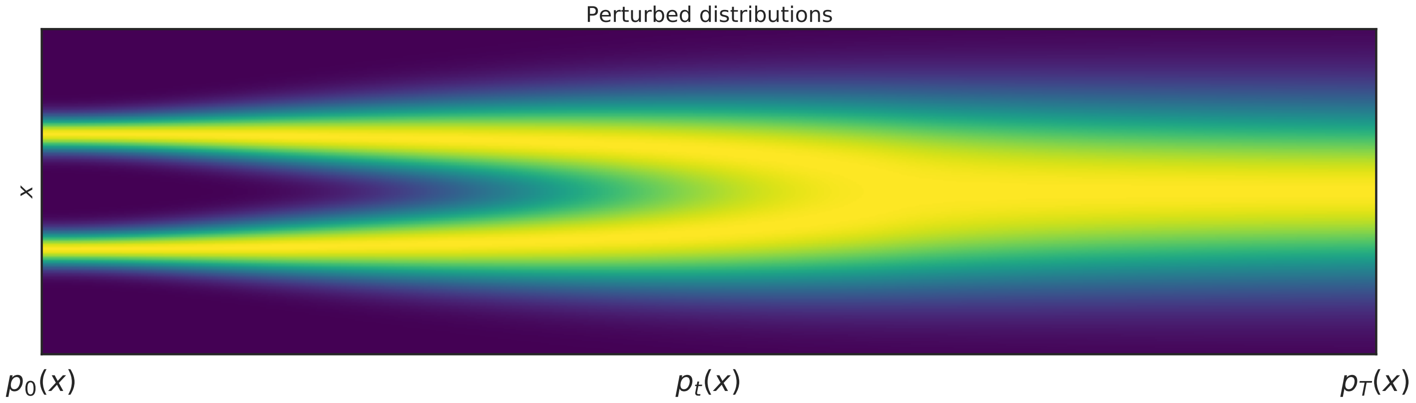


Using an infinite number of noise scales

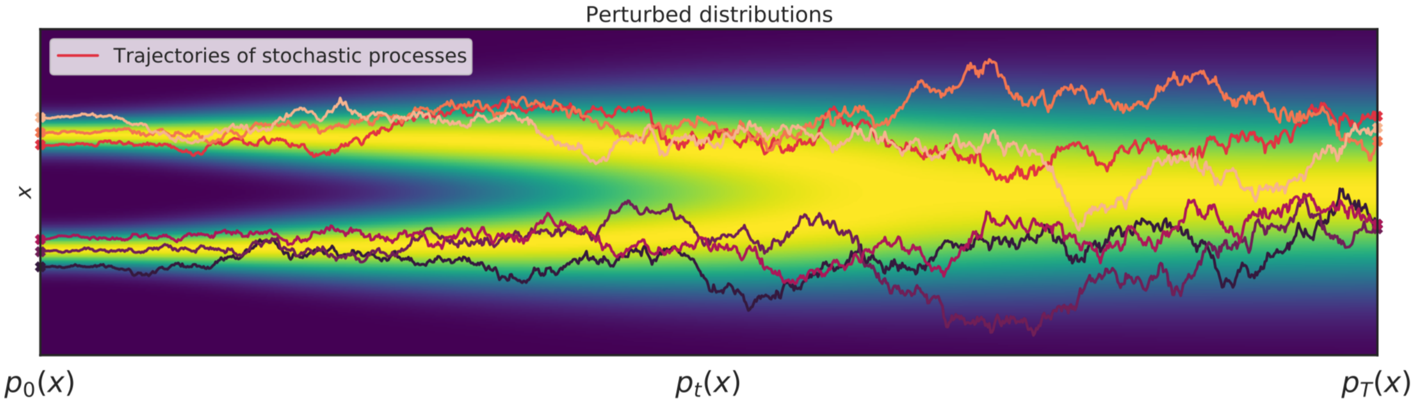


$t \in [0, T]$: continuous index of perturbed distributions

Compact representation of infinite distributions

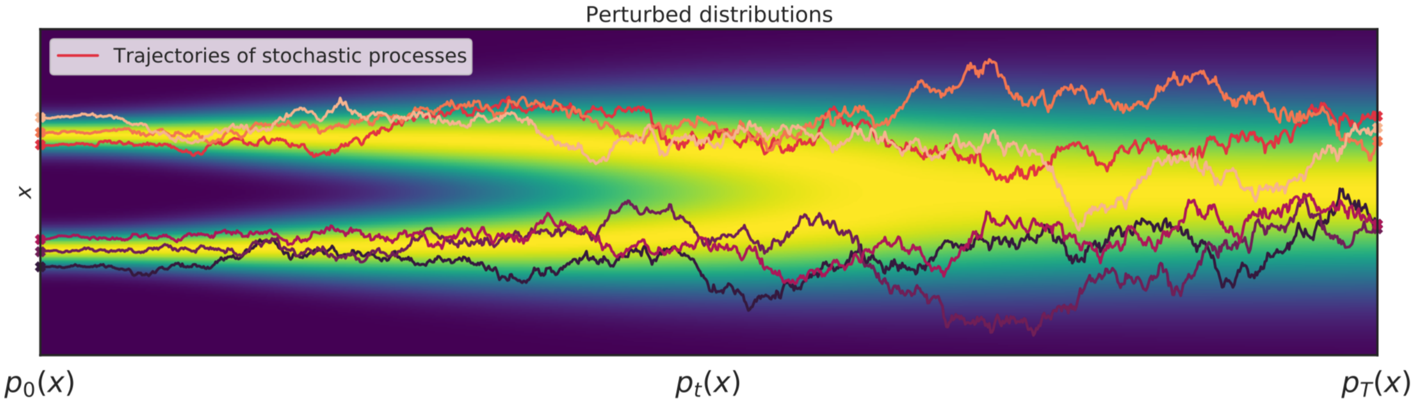


Compact representation of infinite distributions



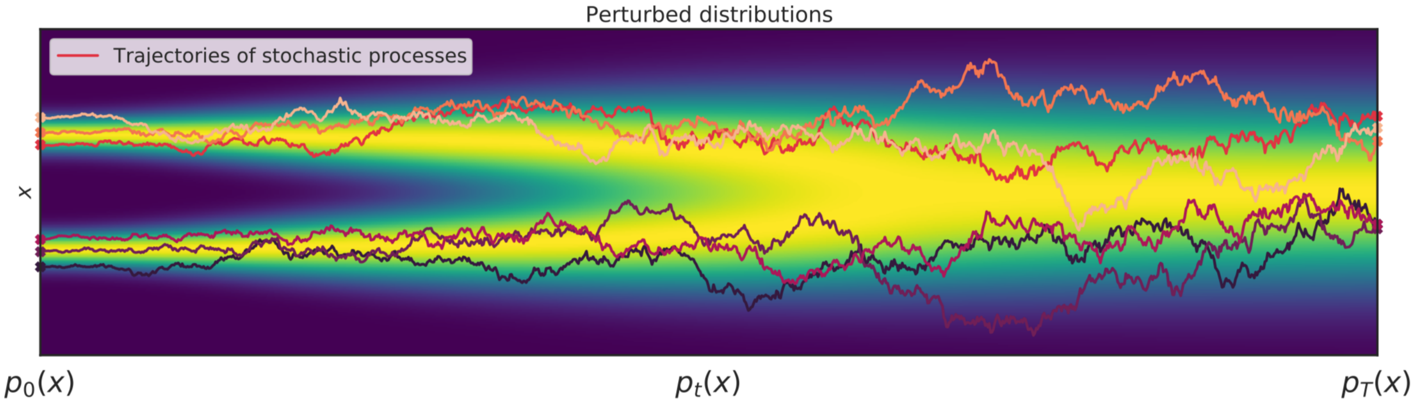
- Stochastic process $\{\mathbf{x}(t)\}_{t=0}^{T}$ → Marginal probability densities $\{p_t(\mathbf{x})\}_{t=0}^{T}$

Compact representation of infinite distributions



- Stochastic process $\{\mathbf{x}(t)\}_{t=0}^{T}$ → Marginal probability densities $\{p_t(\mathbf{x})\}_{t=0}^{T}$
- Stochastic differential equation: $\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + \sigma(t)\mathrm{d}\mathbf{w}$

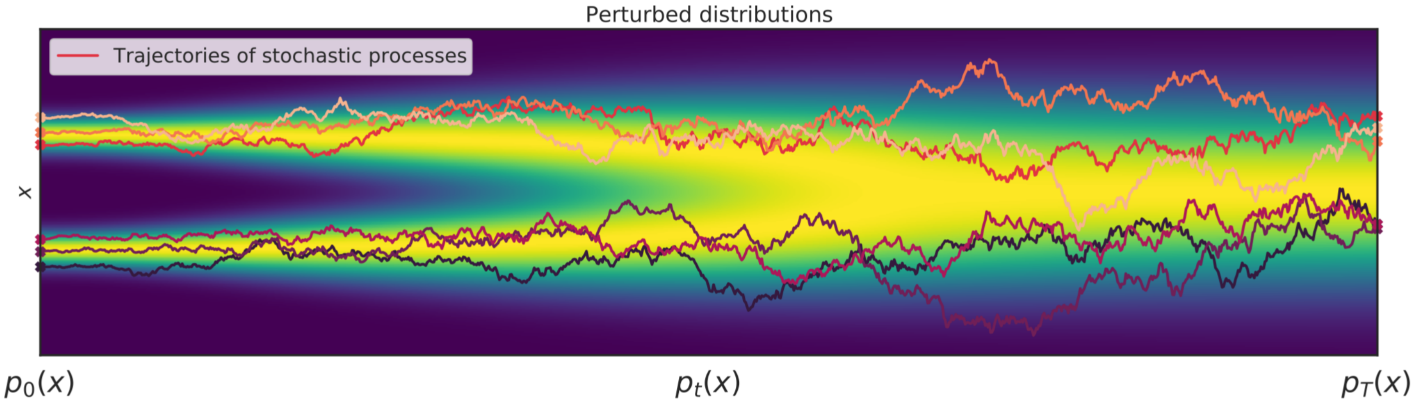
Compact representation of infinite distributions



- Stochastic process $\{\mathbf{x}(t)\}_{t=0}^{T}$ → Marginal probability densities $\{p_t(\mathbf{x})\}_{t=0}^{T}$
- Stochastic differential equation: $\mathrm{d}\mathbf{x} = \boxed{\mathbf{f}(\mathbf{x}, t)\mathrm{d}t} + \sigma(t)\mathrm{d}\mathbf{w}$

Deterministic drift

Compact representation of infinite distributions

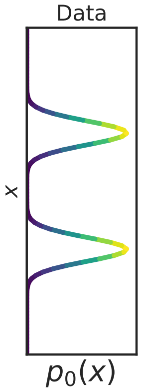


- Stochastic process $\{\mathbf{x}(t)\}_{t=0}^{T}$ → Marginal probability densities $\{p_t(\mathbf{x})\}_{t=0}^{T}$
- Stochastic differential equation: $\mathrm{d}\mathbf{x} = \boxed{\mathbf{f}(\mathbf{x}, t)\mathrm{d}t} + \sigma(t)\boxed{\mathrm{d}\mathbf{w}}$

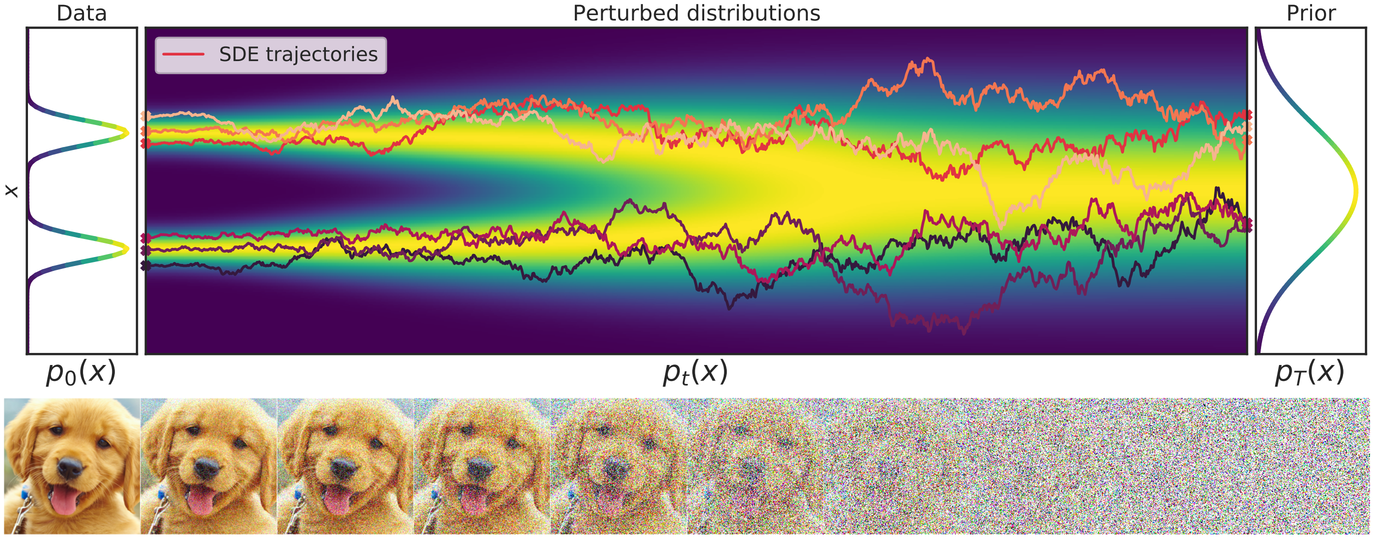
Deterministic drift

Infinitesimal white noise

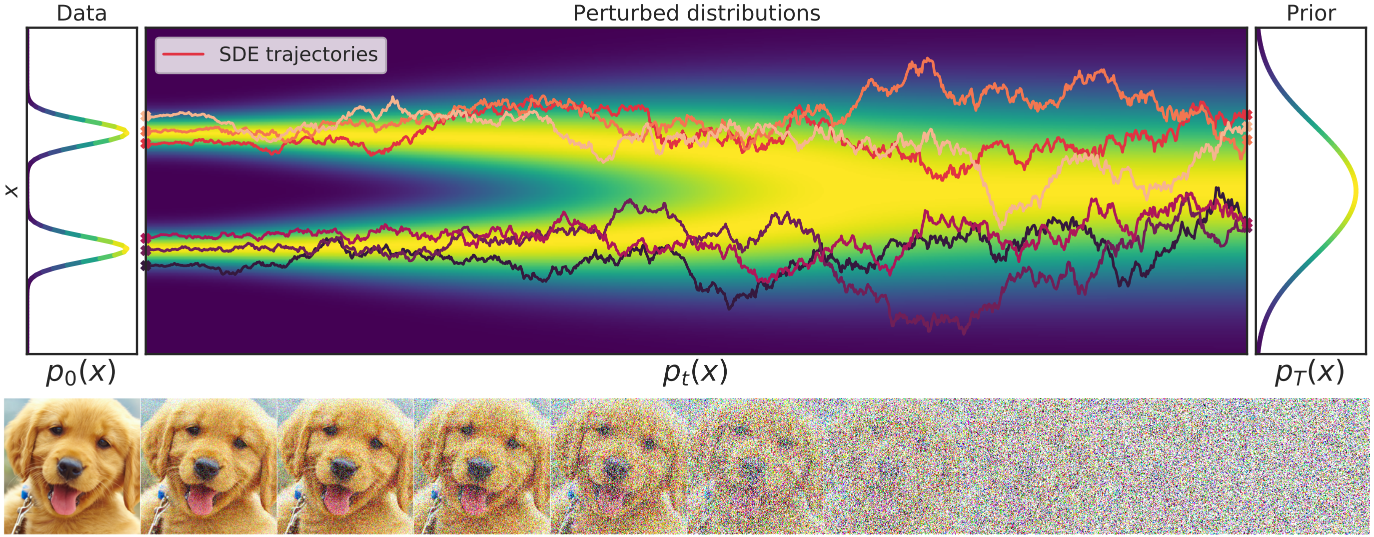
Score-based generative modeling via SDEs



Score-based generative modeling via SDEs

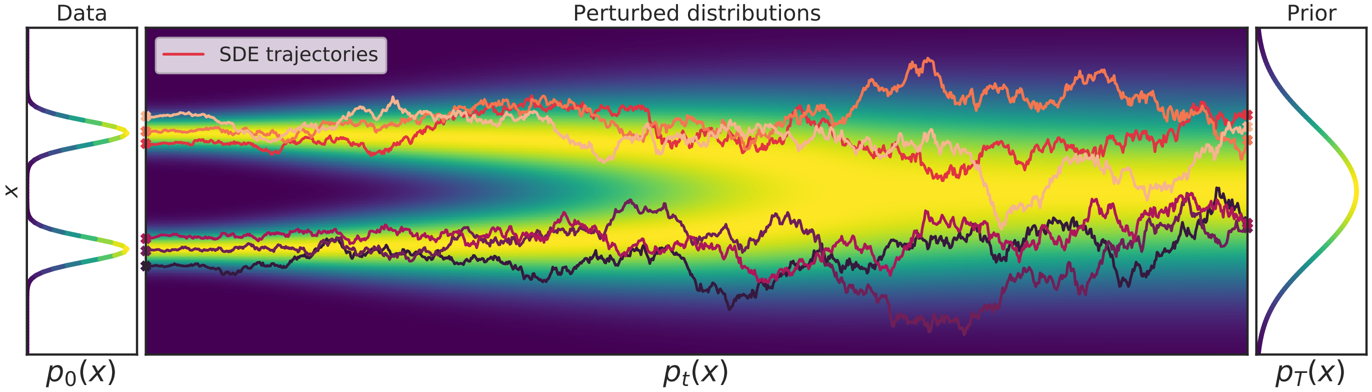


Score-based generative modeling via SDEs

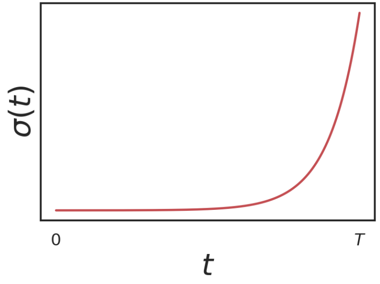


$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w}$$

Score-based generative modeling via SDEs

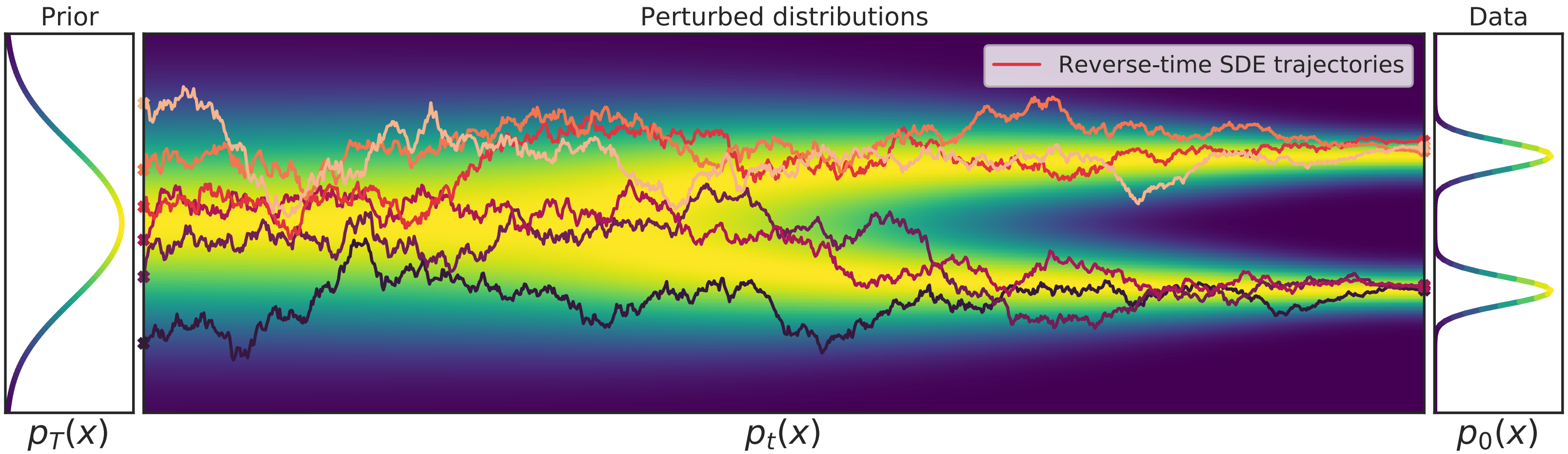


$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w}$$

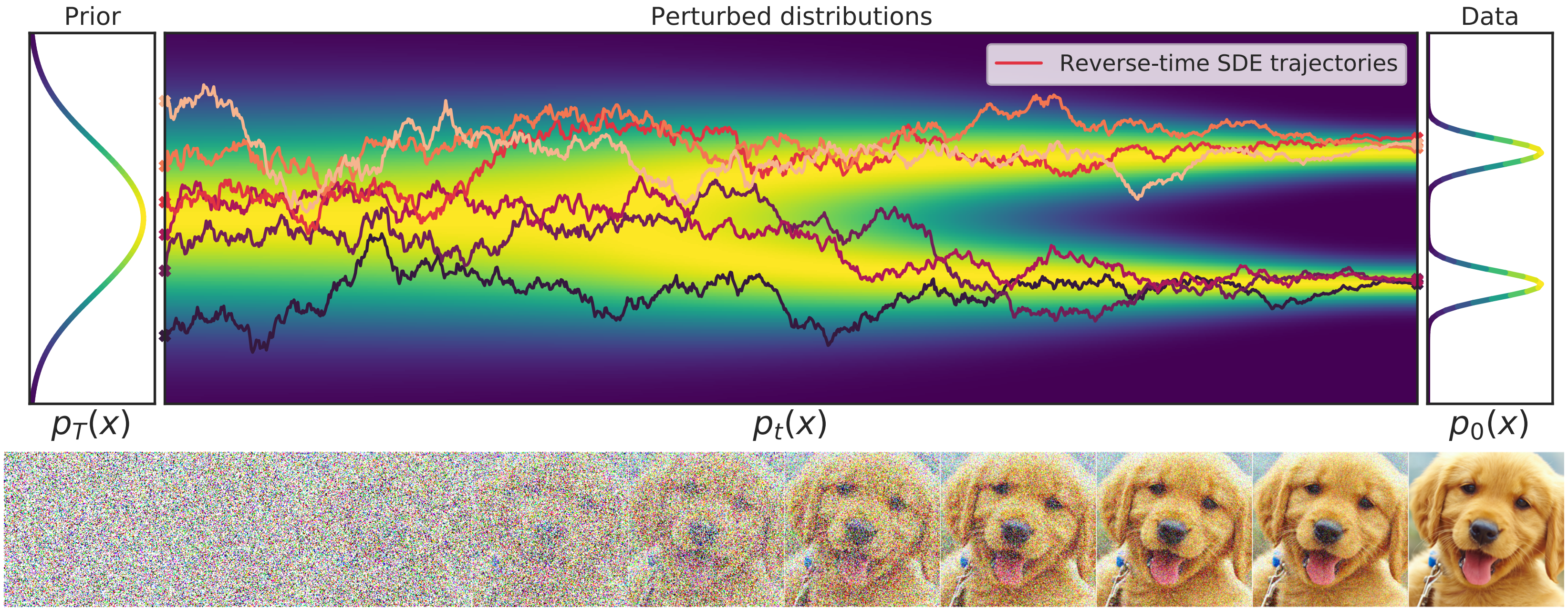


Stanford University

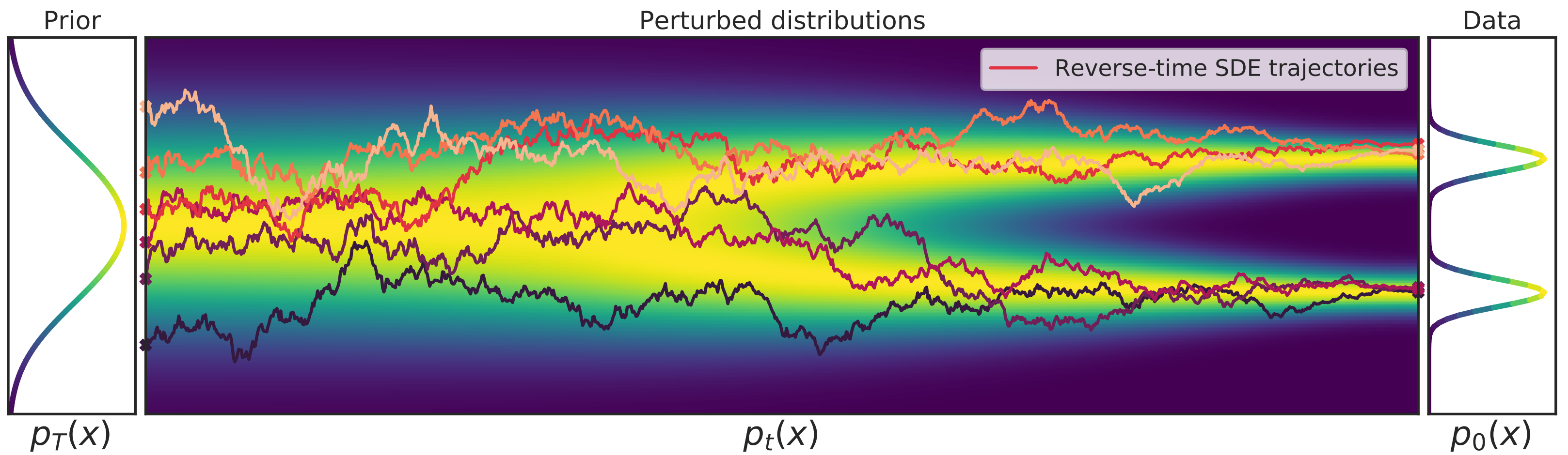
Score-based generative modeling via SDEs



Score-based generative modeling via SDEs



Score-based generative modeling via SDEs



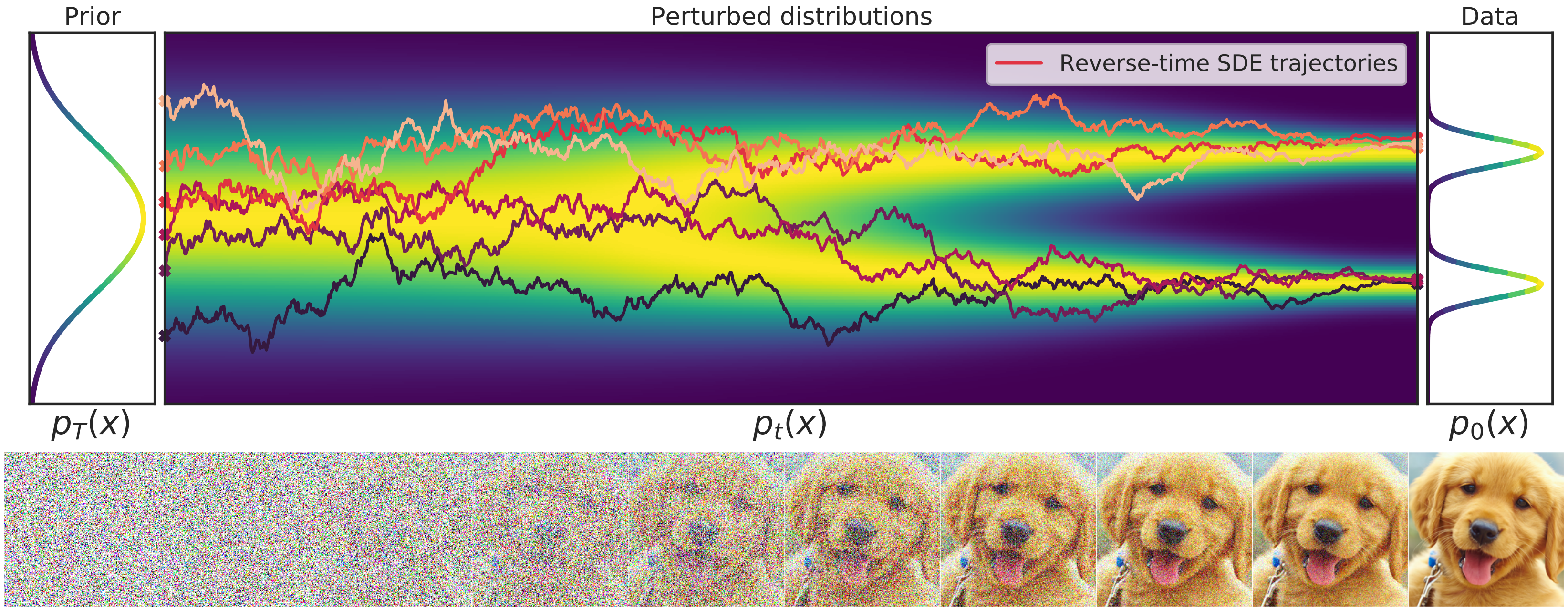
$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w}$$

Time reversal

$\{p_t(\mathbf{x})\}_{t=0}^{T}$

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

Score-based generative modeling via SDEs



$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w}$$

Time reversal

$\{p_t(\mathbf{x})\}_{t=0}^{T}$

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

Score function!

Score-based generative modeling via SDEs

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. “Score-Based Generative Modeling through Stochastic Differential Equations.” ICLR 2021.

Score-based generative modeling via SDEs

- Time-dependent score-based model

$$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Score-based generative modeling via SDEs

- Time-dependent score-based model

$$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$$

- Training:

$$\mathbb{E}_{t \in \mathcal{U}(0,T)}[\lambda(t)\mathbb{E}_{p_t(\mathbf{x})}[||\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)||_2^2]]$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Score-based generative modeling via SDEs

- Time-dependent score-based model

$$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$$

- Training:

$$\mathbb{E}_{t \in \mathcal{U}(0,T)}[\lambda(t)\mathbb{E}_{p_t(\mathbf{x})}[||\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)||_2^2]]$$

- Reverse-time SDE

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Score-based generative modeling via SDEs

- Time-dependent score-based model

$$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$$

- Training:

$$\mathbb{E}_{t \in \mathcal{U}(0,T)}[\lambda(t)\mathbb{E}_{p_t(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\|_2^2]]$$

- Reverse-time SDE

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

- Euler-Maruyama (analgous to Euler for ODEs)

$$\mathbf{x} \leftarrow \mathbf{x} - \sigma(t)^2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\Delta t + \sigma(t)\ \mathbf{z} \quad (\mathbf{z} \sim \mathcal{N}(\mathbf{0}, |\Delta t|\ \mathbf{I}))$$

$$t \leftarrow t + \Delta t$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Mixture of score matching subsumes MLE

$$\mathbb{E}_{t\in\mathcal{U}(0,T)}[\lambda(t)\mathbb{E}_{p_t(\mathbf{x})}[\|\nabla_{\mathbf{x}}\log p_t(\mathbf{x}) - \nabla_{\mathbf{x}}\log q_{t,\boldsymbol{\theta}}(\mathbf{x},t)\|_2^2]]$$

$$=$$

$$\frac{2}{T}\mathrm{KL}(p_0(\mathbf{x}) \parallel q_{0,\boldsymbol{\theta}}(\mathbf{x}))$$

For certain choice of the weighting function $\lambda(t)$

Durkan* and **Song***, “On Maximum Likelihood Training of Score-Based Generative Models”, arXiv 2101.09258

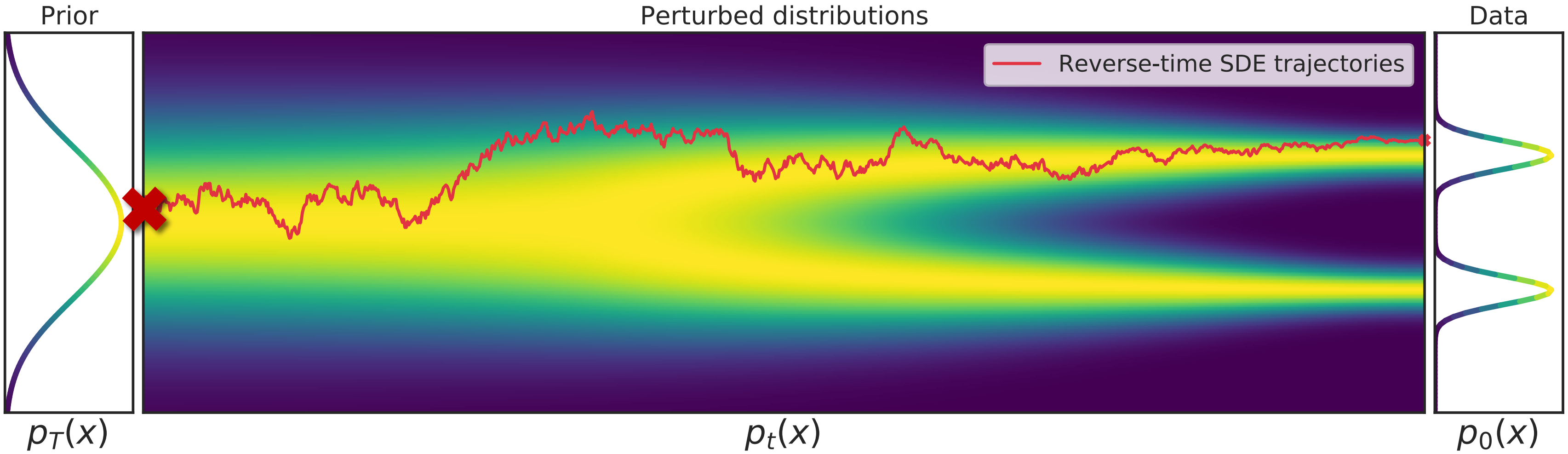
Predictor-Corrector sampling methods

- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. “Score-Based Generative Modeling through Stochastic Differential Equations.” ICLR 2021.

Predictor-Corrector sampling methods

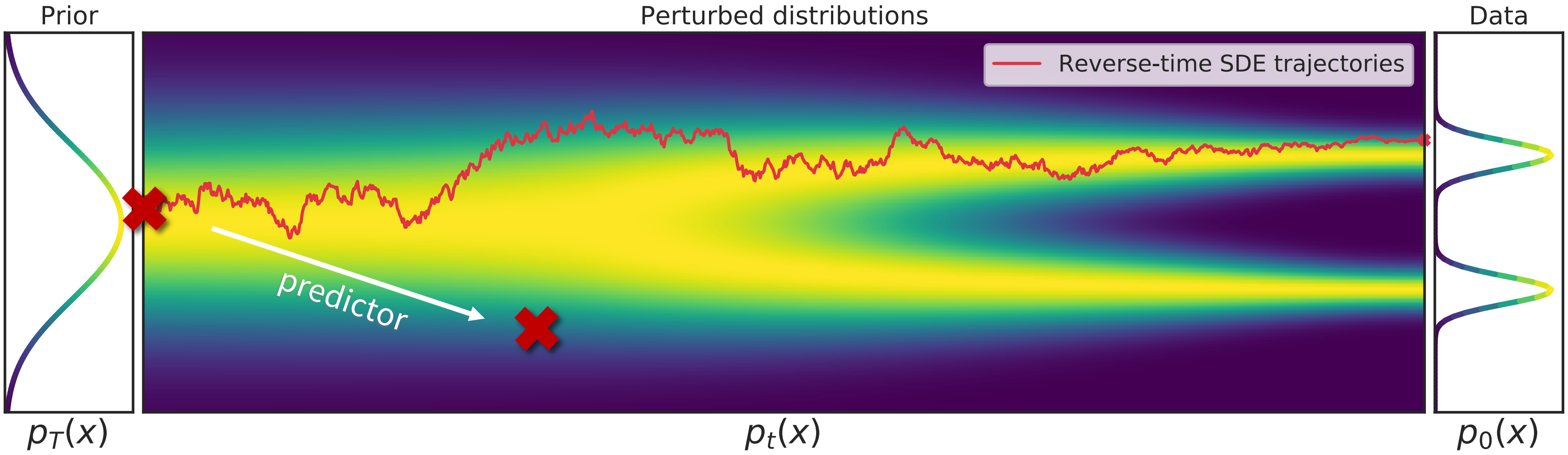
- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Predictor-Corrector sampling methods

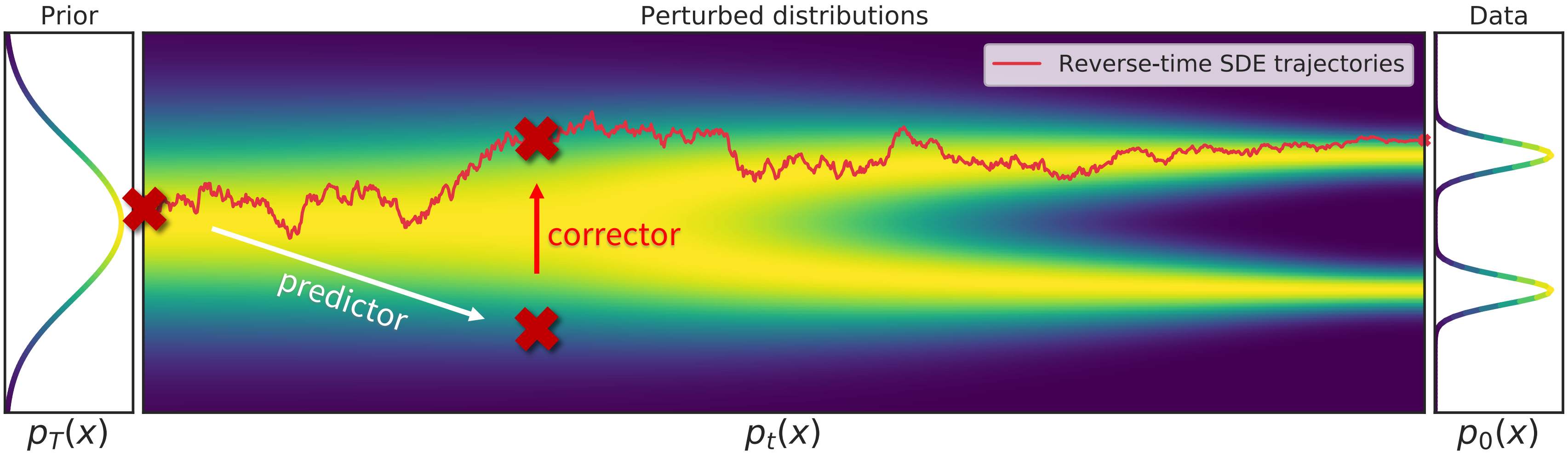
- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Predictor-Corrector sampling methods

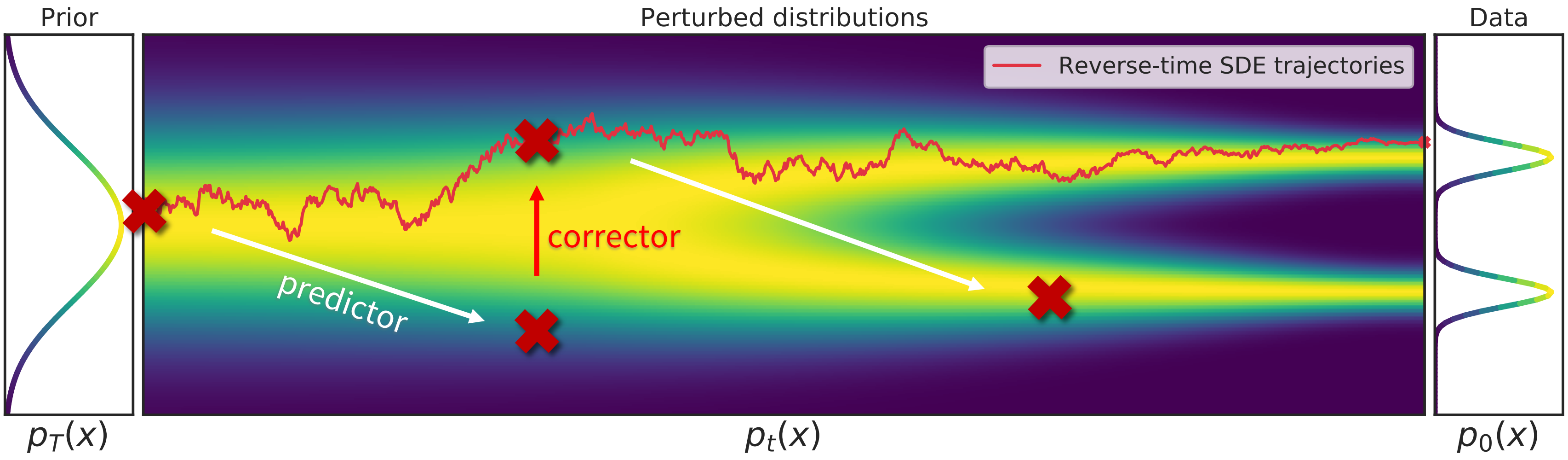
- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Predictor-Corrector sampling methods

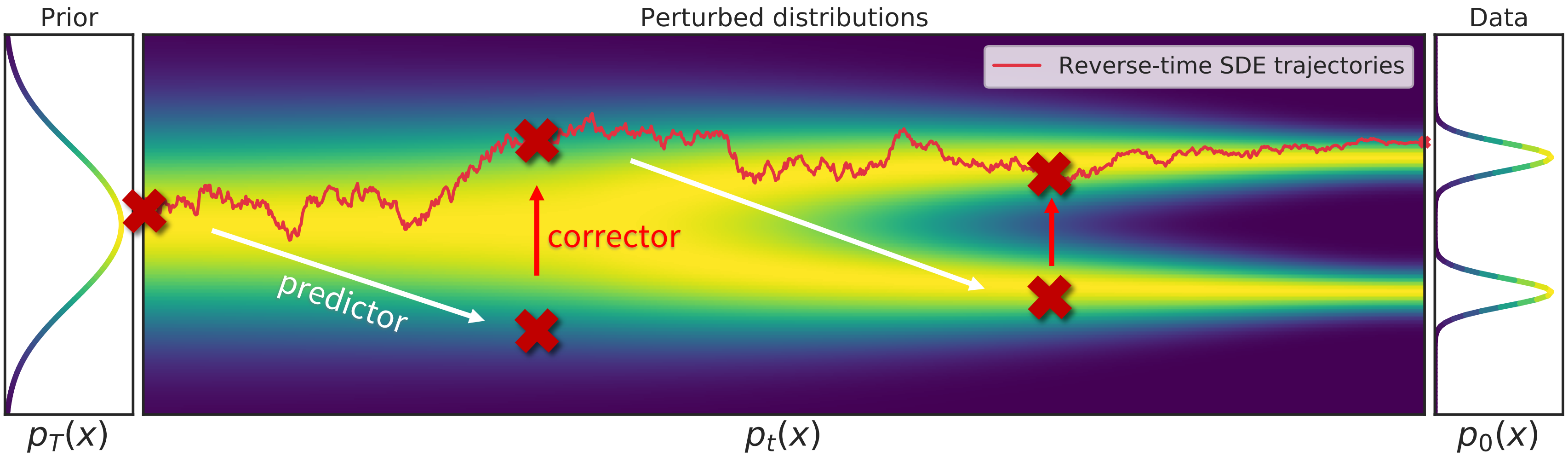
- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Predictor-Corrector sampling methods

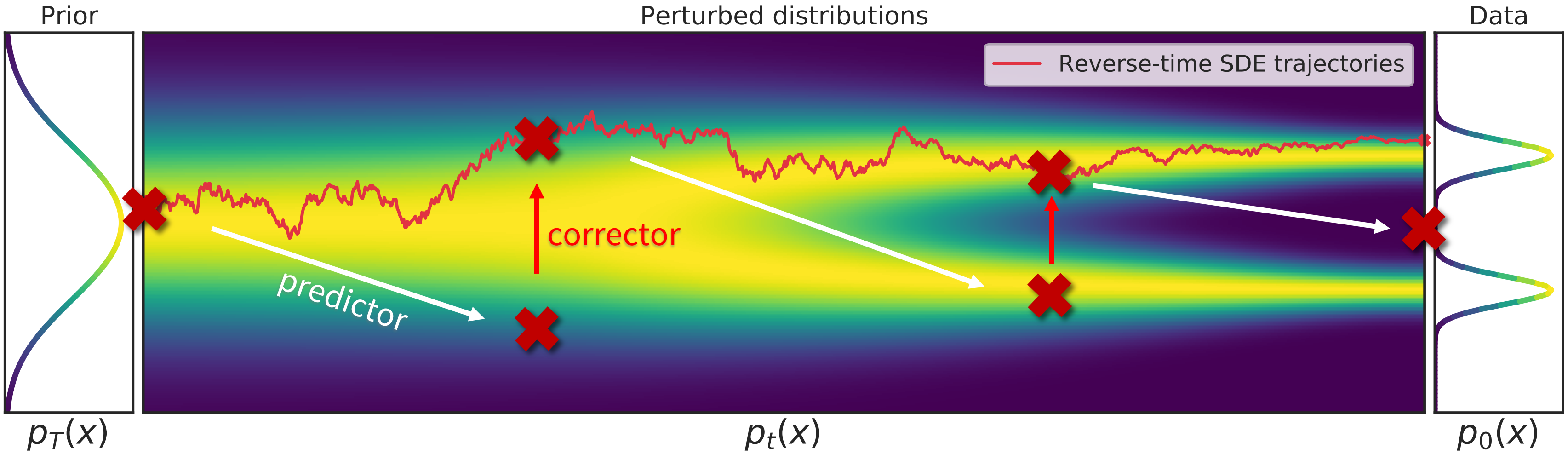
- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Predictor-Corrector sampling methods

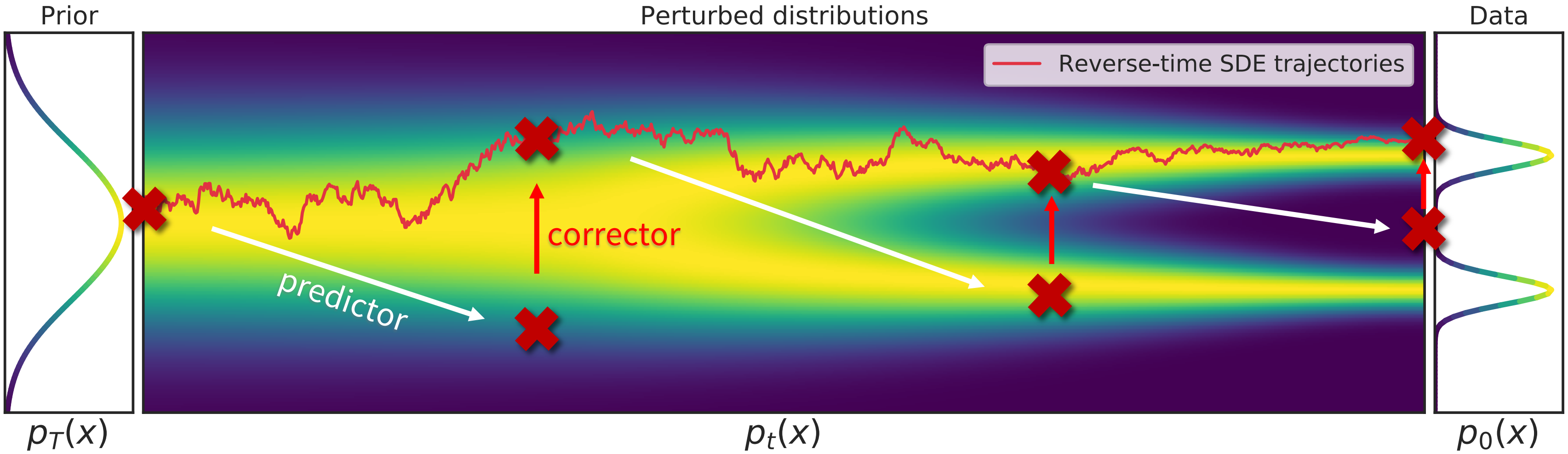
- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Predictor-Corrector sampling methods

- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Results on predictor-corrector sampling

Model	FID↓	IS↑
Conditional		
BigGAN (Brock et al., 2018)	14.73	9.22
StyleGAN2-ADA (Karras et al., 2020a)	**2.42**	**10.14**
Unconditional		
StyleGAN2-ADA (Karras et al., 2020a)	2.92	9.83
NCSN (Song & Ermon, 2019)	25.32	8.87 ± .12
NCSNv2 (Song & Ermon, 2020)	10.87	8.40 ± .07
DDPM (Ho et al., 2020)	3.17	9.46 ± .11
DDPM++	2.78	9.64
DDPM++ cont. (VP)	2.55	9.58
DDPM++ cont. (sub-VP)	2.61	9.56
DDPM++ cont. (deep, VP)	2.41	9.68
DDPM++ cont. (deep, sub-VP)	2.41	9.57
NCSN++	2.45	9.73
NCSN++ cont. (VE)	2.38	9.83
NCSN++ cont. (deep, VE)	**2.20**	**9.89**

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

High-Fidelity Generation for 1024x1024 Images

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: turning the SDE to ODE

- Probability flow ODE (ordinary differential equation)

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. “Score-Based Generative Modeling through Stochastic Differential Equations.” ICLR 2021.

Probability flow ODE: turning the SDE to ODE

- Probability flow ODE (ordinary differential equation)

$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w}$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: turning the SDE to ODE

- Probability flow ODE (ordinary differential equation)

$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w} \quad \underset{\{p_t(\mathbf{x})\}_{t=0}^{T}}{=\!=} \quad \mathrm{d}\mathbf{x} = -\frac{1}{2}\sigma(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \mathrm{d}t$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: turning the SDE to ODE

- Probability flow ODE (ordinary differential equation)

$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w} \quad \underset{\{p_t(\mathbf{x})\}_{t=0}^{T}}{=} \quad \mathrm{d}\mathbf{x} = -\frac{1}{2}\sigma(t)^2 \underbrace{\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}}_{\substack{\text{Score function} \\ \approx \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)}} \mathrm{d}t$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

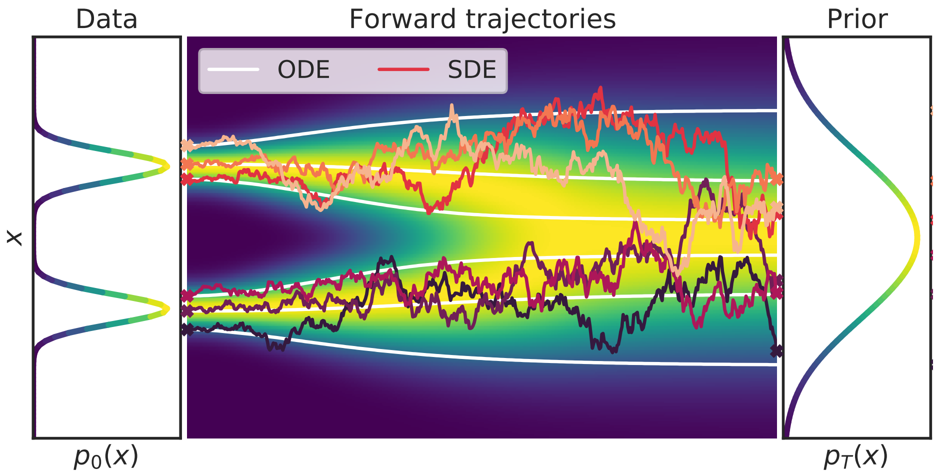
Probability flow ODE: turning the SDE to ODE

- Probability flow ODE (ordinary differential equation)

$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w} \quad \underset{\{p_t(\mathbf{x})\}_{t=0}^{T}}{=} \quad \mathrm{d}\mathbf{x} = -\frac{1}{2}\sigma(t)^2 \boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})} \mathrm{d}t$$

Score function

$\approx \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$



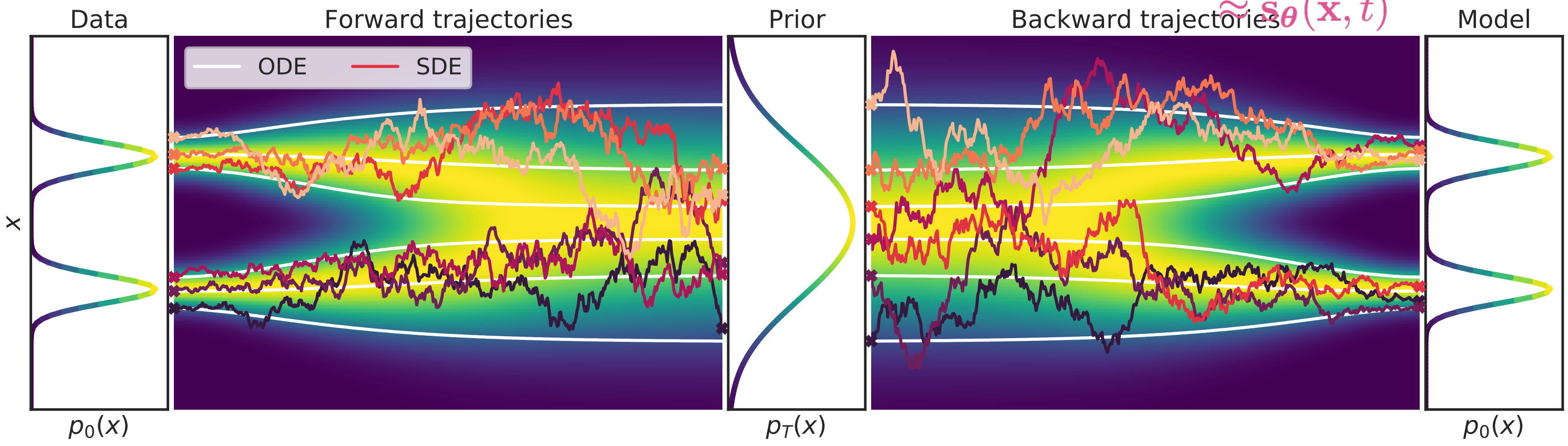
Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: turning the SDE to ODE

- Probability flow ODE (ordinary differential equation)

$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w} \qquad \underset{\{p_t(\mathbf{x})\}_{t=0}^{T}}{=} \qquad \mathrm{d}\mathbf{x} = -\frac{1}{2}\sigma(t)^2 \boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}\mathrm{d}t$$

Score function $\approx \mathbf{s}_\theta(\mathbf{x}, t)$



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

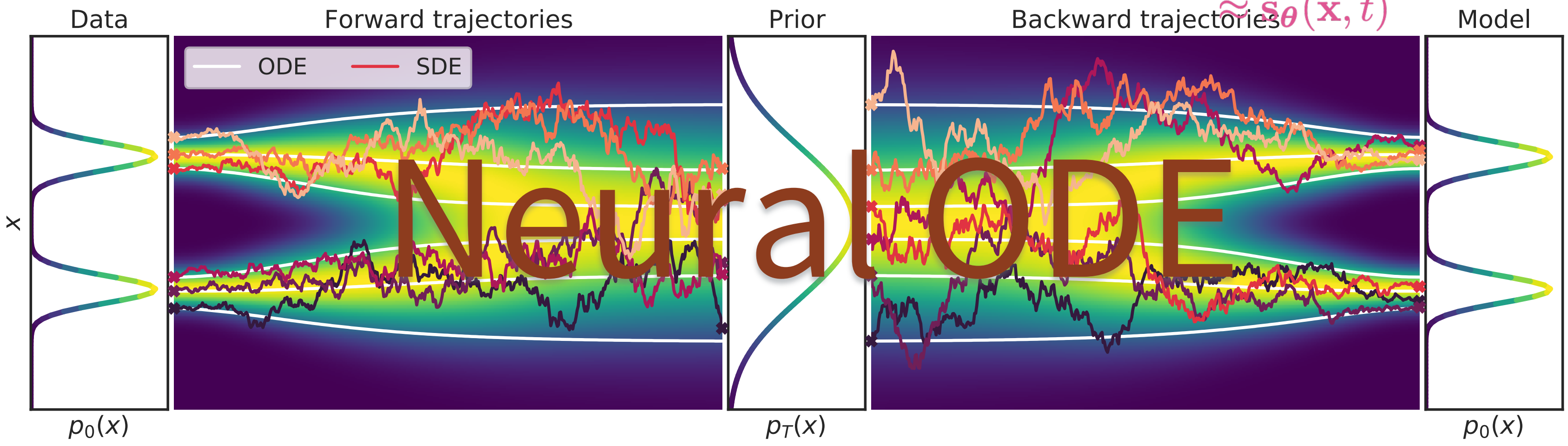
Probability flow ODE: turning the SDE to ODE

- Probability flow ODE (ordinary differential equation)

$$\mathrm{d}\mathbf{x} = \sigma(t)\mathrm{d}\mathbf{w} \quad = \quad \mathrm{d}\mathbf{x} = -\frac{1}{2}\sigma(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \mathrm{d}t$$

$\{p_t(\mathbf{x})\}_{t=0}^{T}$

Score function $\approx \mathbf{s}_\theta(\mathbf{x}, t)$



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. “Score-Based Generative Modeling through Stochastic Differential Equations.” ICLR 2021.

Solving Reverse-ODE for Sampling

- **More efficient samplers** via black-box ODE solvers

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. “Score-Based Generative Modeling through Stochastic Differential Equations.” ICLR 2021.

Solving Reverse-ODE for Sampling

- **More efficient samplers** via black-box ODE solvers

NFE = Number of score Function Evaluations

- Predictor-corrector:
  - $>$ 1000 NFE
- ODE
  - $\approx$100 NFE

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Solving Reverse-ODE for Sampling

- **More efficient samplers** via black-box ODE solvers

NFE = Number of score Function Evaluations

- Predictor-corrector:
  - $>$ 1000 NFE
- ODE
  - $\approx$100 NFE

- **Exact likelihood** though models are trained with score matching.

$$\log p_0(\mathbf{x}) = \log p_T(\mathbf{x}) - \frac{1}{2}\int_0^T \sigma(t)^2 \mathrm{trace}(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t))\mathrm{d}t$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. “Score-Based Generative Modeling through Stochastic Differential Equations.” ICLR 2021.

Solving Reverse-ODE for Sampling

Model	NLL Test ↓	FID ↓
RealNVP (Dinh et al., 2016)	3.49	-
iResNet (Behrmann et al., 2019)	3.45	-
Glow (Kingma & Dhariwal, 2018)	3.35	-
MintNet (Song et al., 2019b)	3.32	-
Residual Flow (Chen et al., 2019)	3.28	46.37
FFJORD (Grathwohl et al., 2018)	3.40	-
Flow++ (Ho et al., 2019)	3.29	-
DDPM (L) (Ho et al., 2020)	$\leqslant 3.70^{*}$	13.51
DDPM (L_{simple}) (Ho et al., 2020)	$\leqslant 3.75^{*}$	3.17
DDPM	3.28	3.37
DDPM cont. (VP)	3.21	3.69
DDPM cont. (sub-VP)	3.05	3.56
DDPM++ cont. (VP)	3.16	3.93
DDPM++ cont. (sub-VP)	3.02	3.16
DDPM++ cont. (deep, VP)	3.13	3.08
DDPM++ cont. (deep, sub-VP)	**2.99**	**2.92**

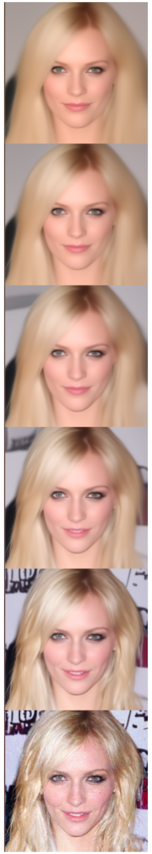
models trained
with score matching

black-box ODE
Solvers for sampling

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: latent space manipulation

Interpolation

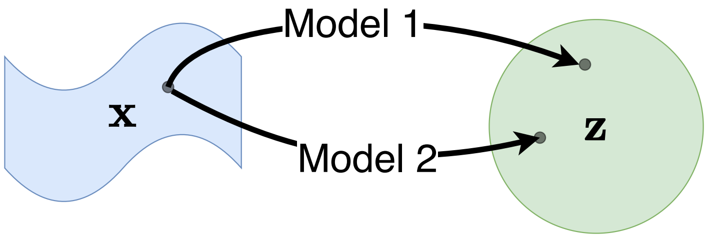


Temperature scaling

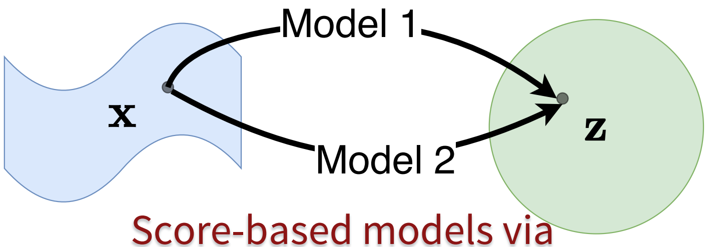
Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: uniquely identifiable encoding

- Uniquely **identifiable** encoding



Flow models, VAE, etc

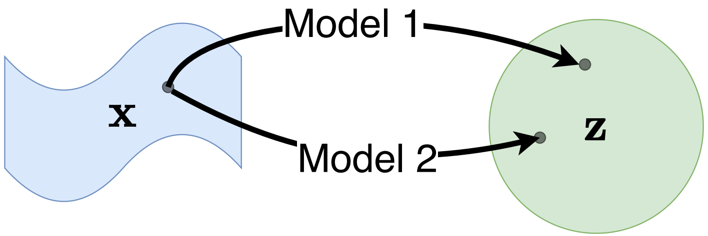


Score-based models via probability flow ODE

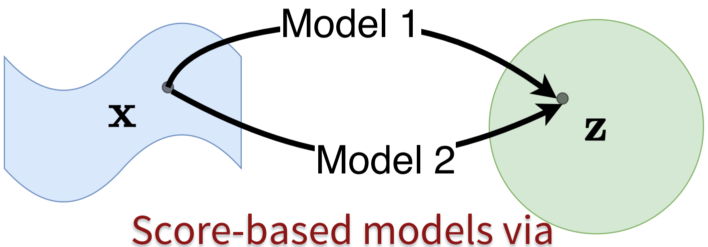
Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: uniquely identifiable encoding

- Uniquely **identifiable** encoding



Flow models, VAE, etc



Score-based models via probability flow ODE

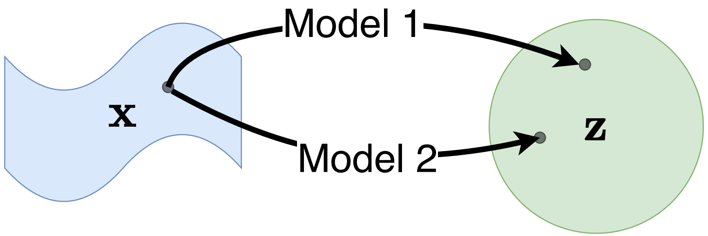
- No trainable parameters in the probability flow ODE!

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\sigma(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \mathrm{d}t$$

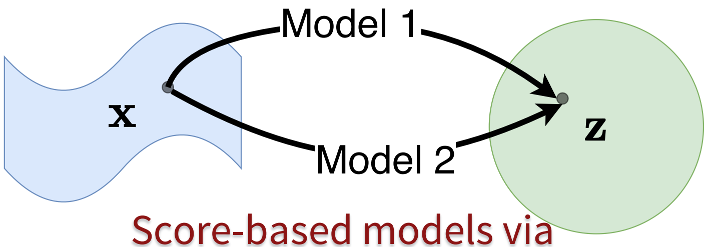
Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Probability flow ODE: uniquely identifiable encoding

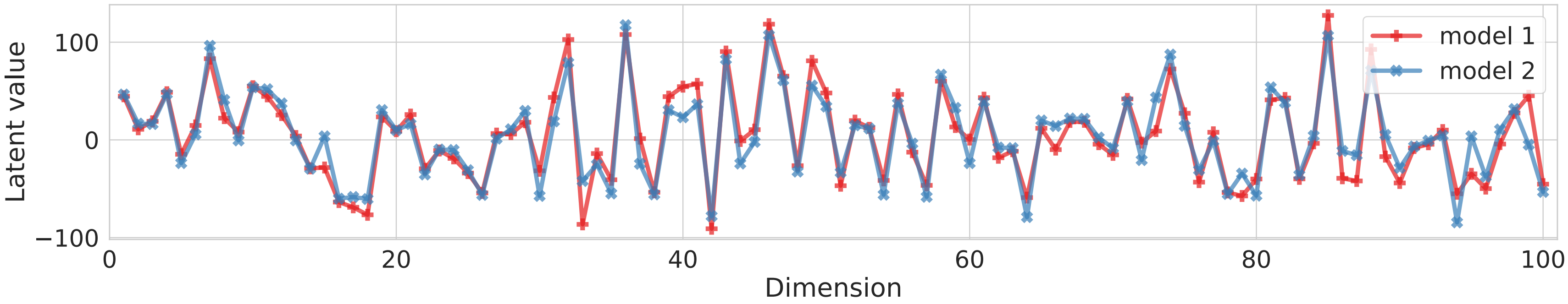
- Uniquely **identifiable** encoding



Flow models, VAE, etc

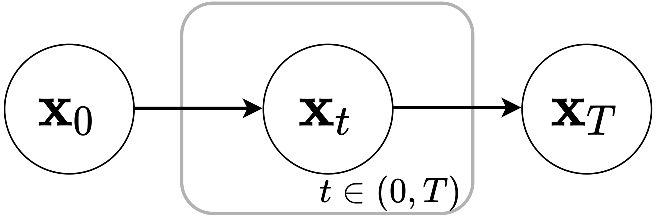


Score-based models via probability flow ODE



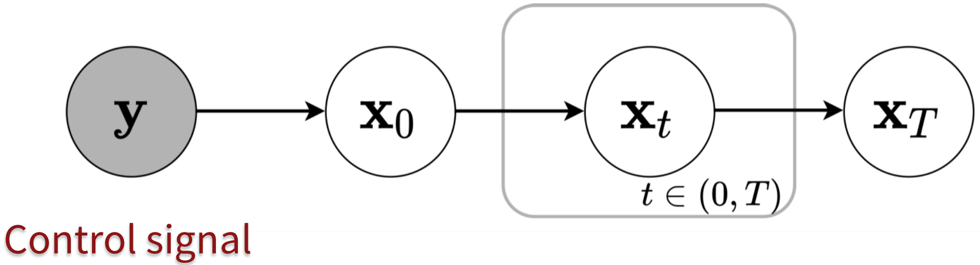
Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation



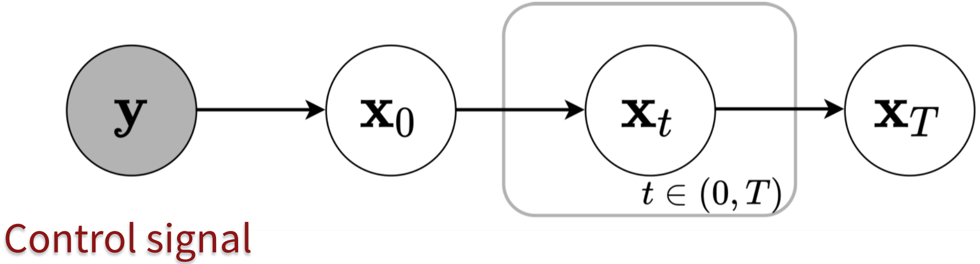
Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation



Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. “Score-Based Generative Modeling through Stochastic Differential Equations.” ICLR 2021.

Controllable Generation

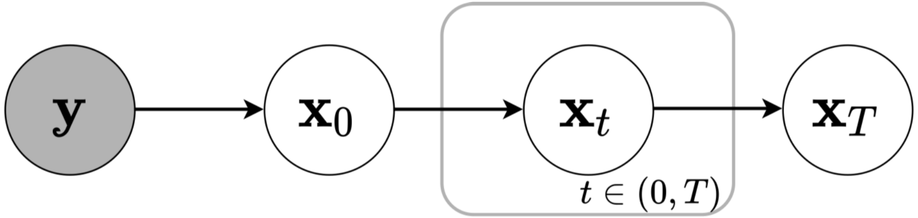


- Conditional reverse-time SDE via unconditional scores

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x} \mid \mathbf{y})\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation



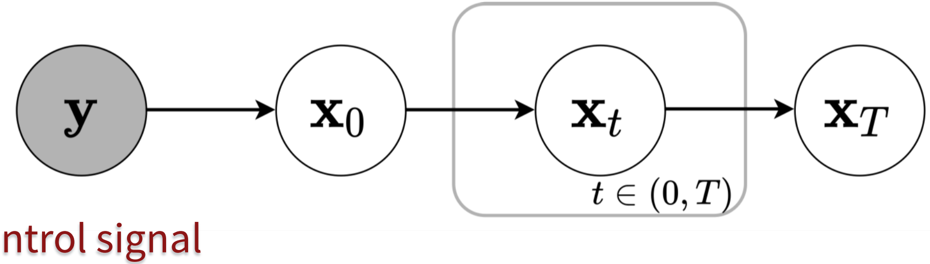
- Conditional reverse-time SDE via unconditional scores

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x} \mid \mathbf{y})\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

?

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation



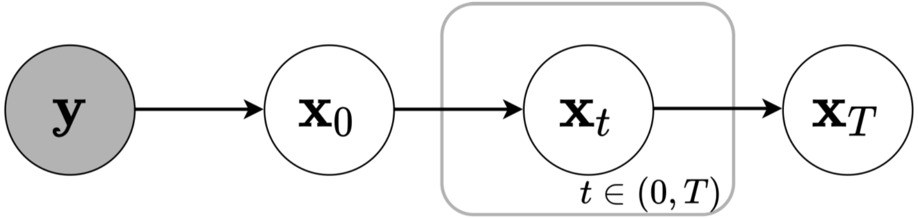
- Conditional reverse-time SDE via unconditional scores

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} \mid \mathbf{y})}\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)[\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})} + \boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{y} \mid \mathbf{x})}]\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation



Control signal

- Conditional reverse-time SDE via unconditional scores

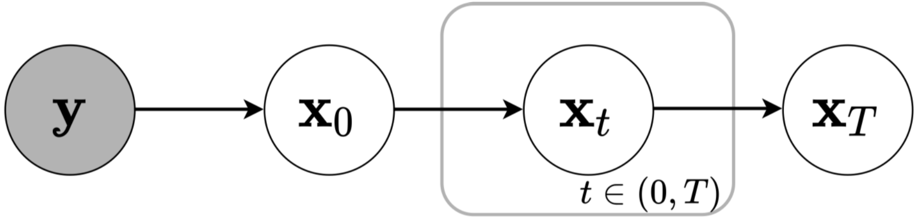
$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} \mid \mathbf{y})}\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)[\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})} + \boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{y} \mid \mathbf{x})}]\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

unconditional score,
Trained w/o y

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation



Control signal

- Conditional reverse-time SDE via unconditional scores

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x} \mid \mathbf{y})}\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)[\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})} + \boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{y} \mid \mathbf{x})}]\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

unconditional score,
Trained w/o y

Trained separately or
specified with domain knowledge

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation: class-conditional generation

- $\mathbf{y}$ is the **class label**
- $p_t(\mathbf{y} \mid \mathbf{x})$ is a time-dependent classifier

class: bird

class: deer

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation: inpainting

- $\mathbf{y}$ is the **masked image**
- $p_t(\mathbf{y} \mid \mathbf{x})$ can be approximated without training

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Controllable Generation: colorization

- $\mathbf{y}$ is the **gray-scale image**
- $p_t(\mathbf{y} \mid \mathbf{x})$ can be approximated without training

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Future directions

- Discrete data, such as text generation
- Theoretical understanding on sample quality
- Faster sampling

Improvements

- Semi-supervised learning
- Inverse problems
- Unrestricted adversarial attacks
- Outlier detection

Applications

Conclusion

Conclusion

- **Gradients of distributions (scores) can be estimated easily**

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible
  - **Stable training** — no minimax optimization

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible
  - **Stable training** — no minimax optimization

- **Better or comparable sample quality to GANs**

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible
  - **Stable training** — no minimax optimization

- **Better or comparable sample quality to GANs**
  - State-of-the-art performance on CIFAR-10 and others

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible
  - **Stable training** — no minimax optimization

- **Better or comparable sample quality to GANs**
  - State-of-the-art performance on CIFAR-10 and others
  - Scalable to resolution of 1024x1024 for image generation

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible
  - **Stable training** — no minimax optimization
- **Better or comparable sample quality to GANs**
  - State-of-the-art performance on CIFAR-10 and others
  - Scalable to resolution of 1024x1024 for image generation
- **Exact likelihood computation**

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible
  - **Stable training** — no minimax optimization
- **Better or comparable sample quality to GANs**
  - State-of-the-art performance on CIFAR-10 and others
  - Scalable to resolution of 1024x1024 for image generation
- **Exact likelihood computation**
  - Competitive likelihood on CIFAR-10

Conclusion

- **Gradients of distributions (scores) can be estimated easily**
  - **Flexible architecture choices** — no need to be normalized/invertible
  - **Stable training** — no minimax optimization
- **Better or comparable sample quality to GANs**
  - State-of-the-art performance on CIFAR-10 and others
  - Scalable to resolution of 1024x1024 for image generation
- **Exact likelihood computation**
  - Competitive likelihood on CIFAR-10
  - Equivalence to Neural ODEs, plus uniquely identifiable encoding

Collaborators

Stefano Ermon

Ben Poole

Jascha Sohl-Dickstein

Durk Kingma

Sahaj Garg

Jiaxin Shi

Abhishek Kumar