MAX PLANCK INSTITUTE
FOR INTELLIGENT SYSTEMS

Overview on Deep Generative Modeling in the Natural Sciences

Introduction

Annalena Kofler, 16.04.2024

Short Introduction: Who am I?

- PhD student at the Max-Planck-Institute for Intelligent Systems in Tübingen supervised by Prof. Bernhard Schölkopf

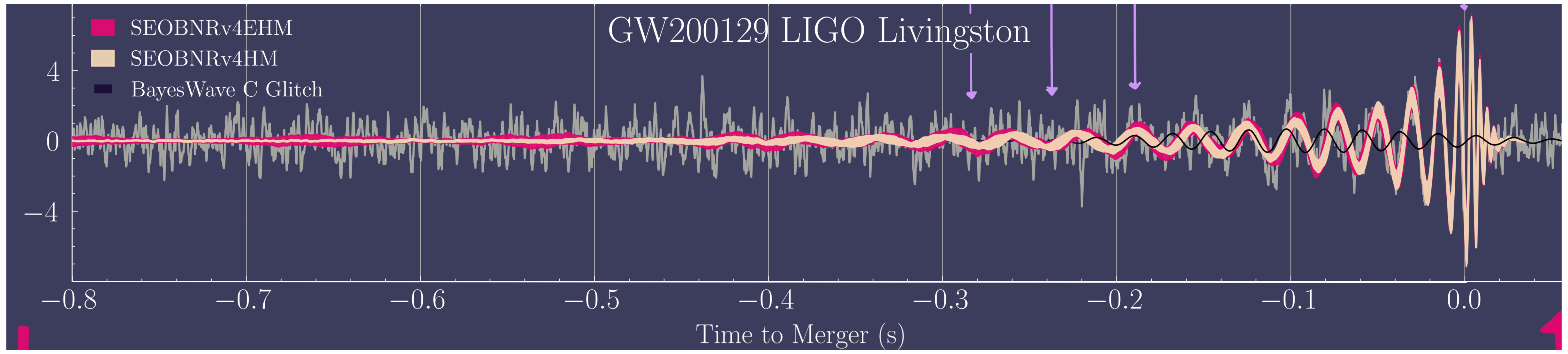
- Working on ML for Physics

    - Posterior Estimation for Gravitational Wave Inference

    - Deep Generative Modeling & Simulation-based inference

    - Differentiable Physics Simulators

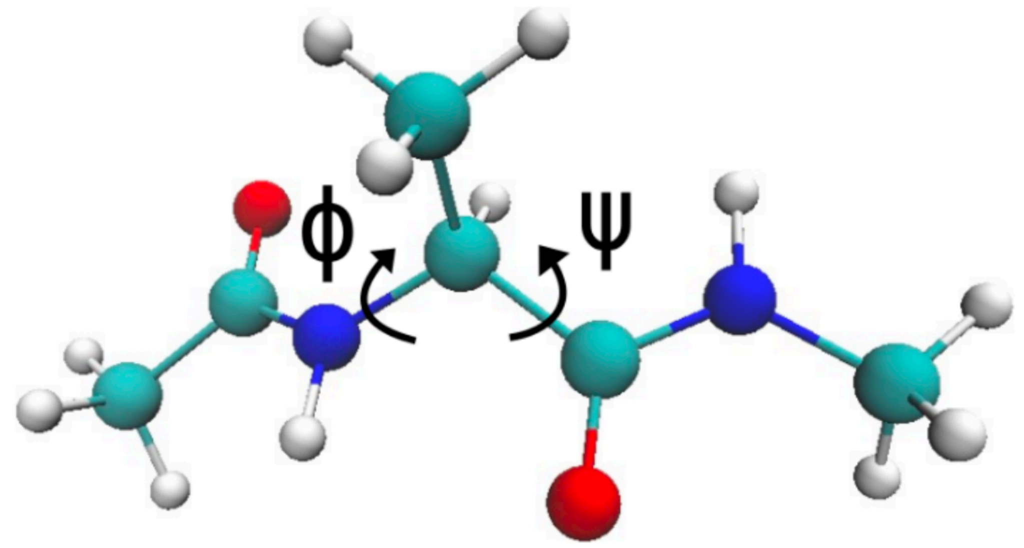
- Previously: Master's student at TUM working with Prof. Lukas Heinrich



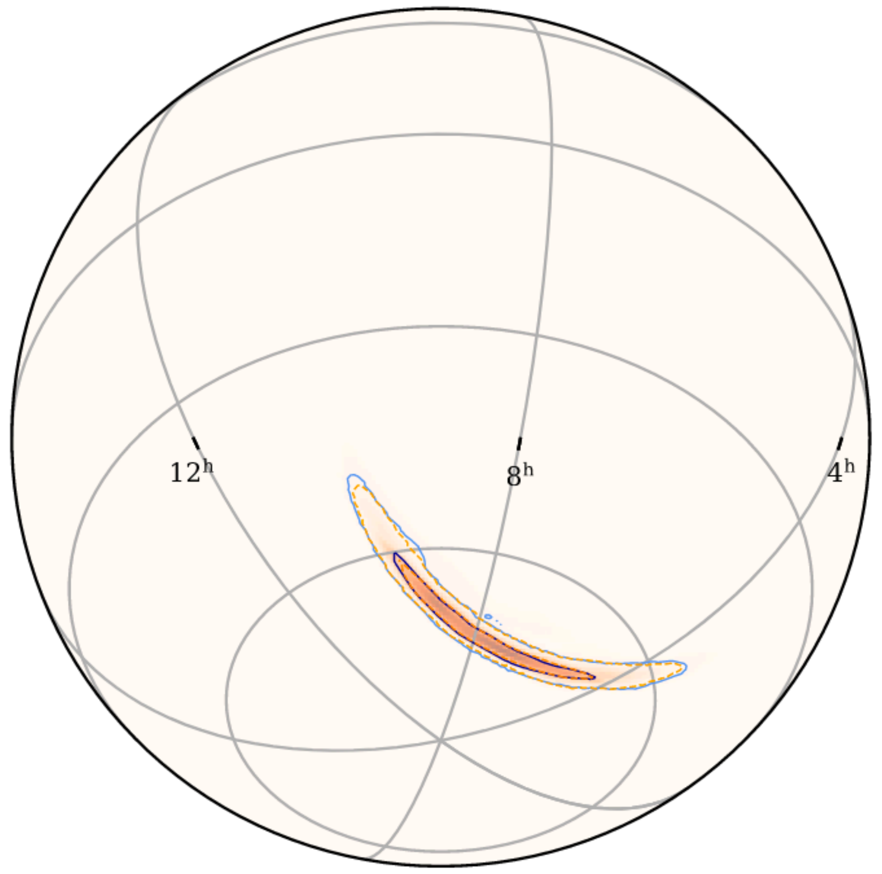
Gupte et al. In preparation, 2024

Why care about probability distributions?

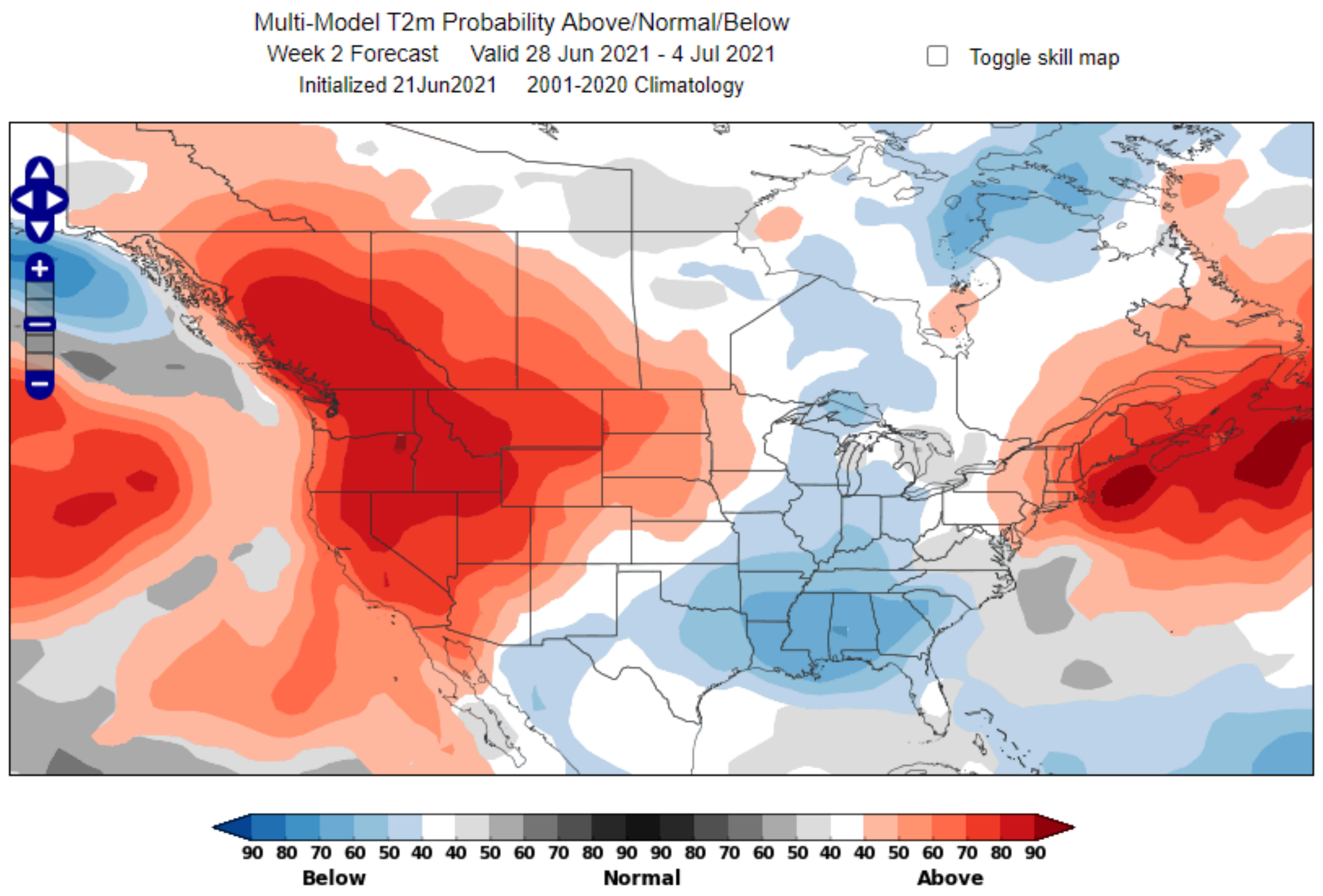
- Data samples usually originate from some distribution
- Examples in natural sciences:
  - Positions of atoms in molecules
  - Parameters of gravitational waves
  - Temperature map
  - …
- But: Distributions might be unknown, e.g. distribution over pixels in image



Midgley, Stimper, et al., Flow Annealed Importance Sampling Bootstrap, ICLR 2023



Dax, et al.,Real-time gravitational-wave science with neural posterior estimation, PRL2021



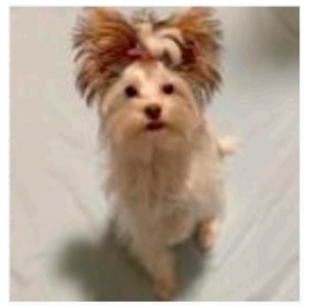
World Climate Service Blog
"What is a probability forecast"

How can we learn a distribution over data?

- Train model to get

model distribution $p_{\text{model}} \approx$ target distribution p_{target}

- What can we do with such models?
    - Sample new data points $x \sim p_{\text{model}}(x)$



$\sim p_{\text{dog-model}}(x)$

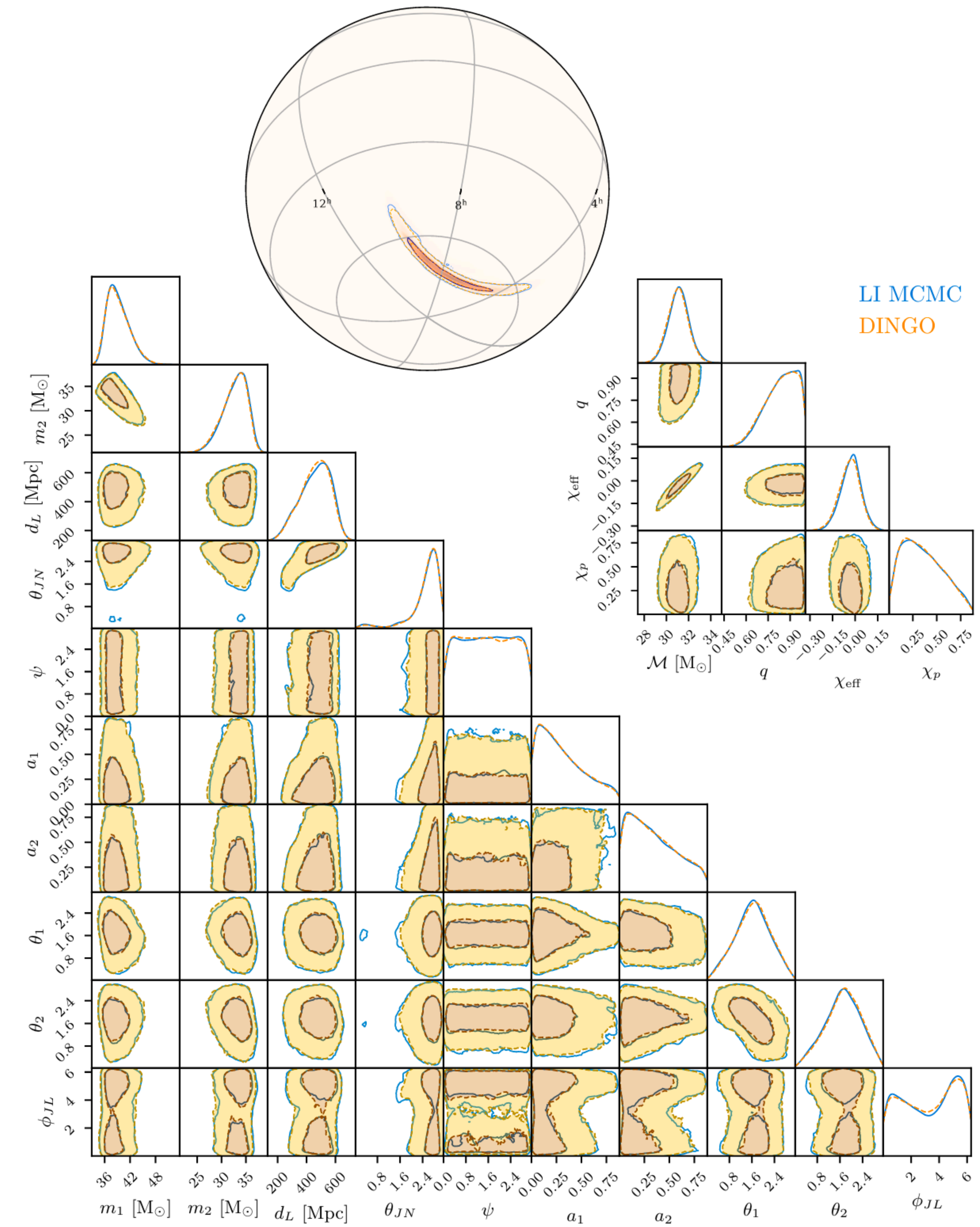
    - Evaluation the model distribution $p_{\text{model}}(x_{\text{sample}})$ for a given data point x_{sample}

$p_{\text{dog-model}}\Big($  $\Big) = 0.87$

Challenge: Complexity of Data Distribution

- Data distributions are usually extremely complex
  - Multi-modal
  - Unknown structure
  - High dimensional, e.g. pixels in image

→ powerful model required



Dax, et al.,Real-time gravitational-wave science with neural posterior estimation, PRL2021

How to build a powerful model?

- Use a Deep Neural Network (DNN) f_θ

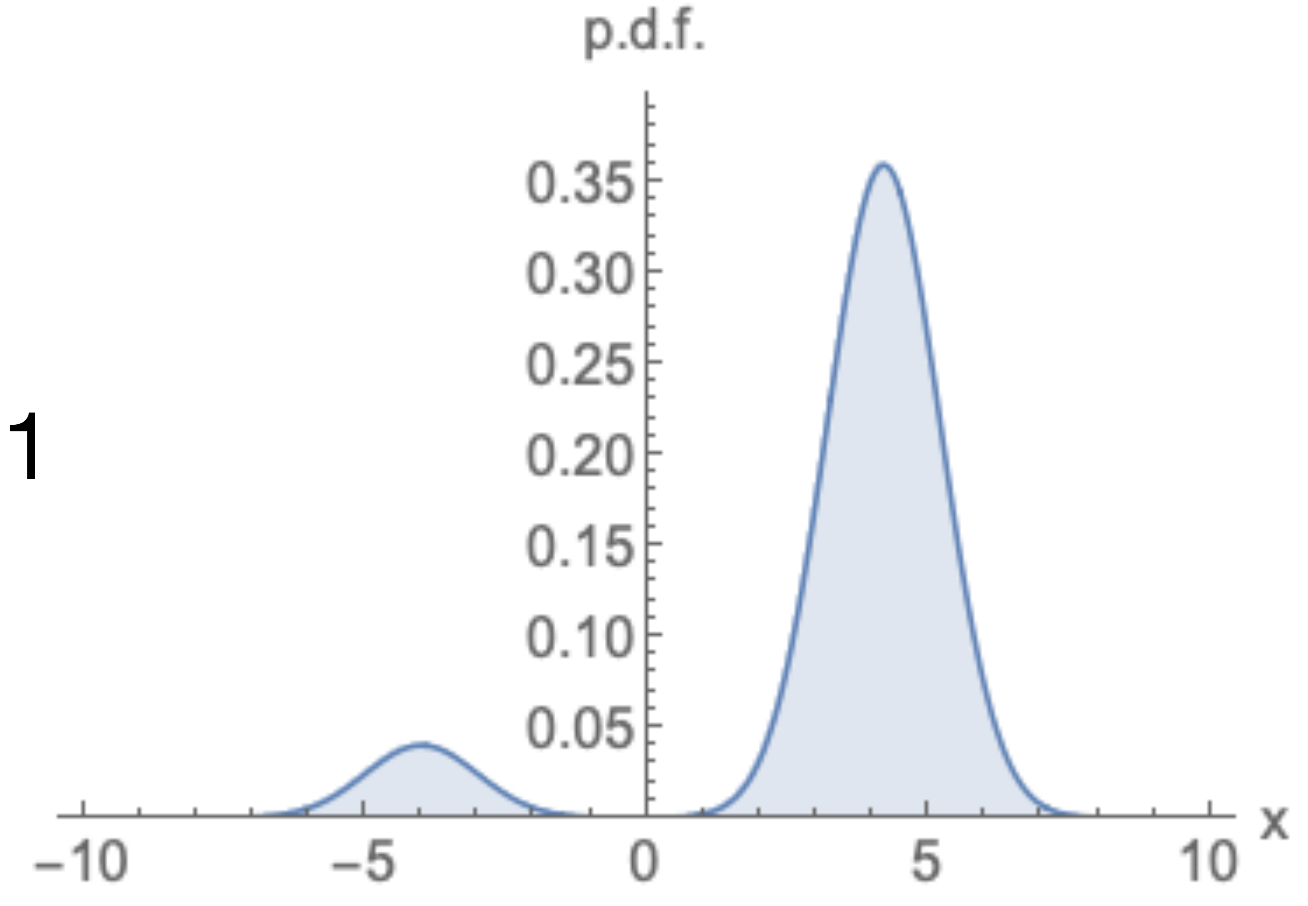
- Is the output $f_\theta(x)$ of the DNN for a sample x already a probability?

$\rightarrow$ No, it does not fulfill the properties of a pdf.

1. Positive: $p(x) \geq 0 \, \forall x$

2. Normalized: $\int p(x) \, \mathrm{d}x = 1$

Area = 1



Animations from blog post by Yang Song

How to build a powerful model?

- How can we adapt $f_\theta(x)$ such that it is a pdf?

  1. Positive → $\exp(f_\theta(x))$
  2. Normalized → $q_\theta(x) = \dfrac{\exp(f_\theta(x))}{Z_\theta}$

     with normalization constant $Z_\theta = \int \exp(f_\theta(x))\mathrm{d}x$

- Problem: calculating the normalization constant intractable for DNN

How to normalize? - Three Model Families

1. Approximate Z_θ
   - Energy-based models

X Inaccurate probability

2. Use restricted NN architectures
   - Variational autoencoders
   - Autoregressive models
   - Normalizing Flows

X Limited flexibility of the model

3. Model the generative process only
   - GANs
   - Diffusion models

X No/approximate evaluation of the probability

How to normalize? - Three Model Families

1. Approximate Z_θ
   - Energy-based models
2. Use restricted NN architectures
   - Variational autoencoders
   - Autoregressive models
   - **Normalizing Flows**
3. Model the generative process only
   - GANs
   - **Diffusion models**

X Inaccurate probability

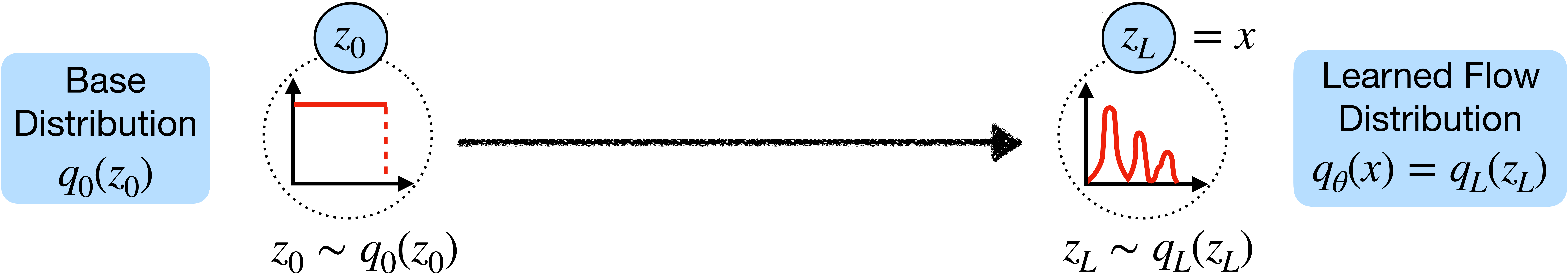
X Limited flexibility of the model

X No/approximate evaluation of the probability

Normalizing Flows

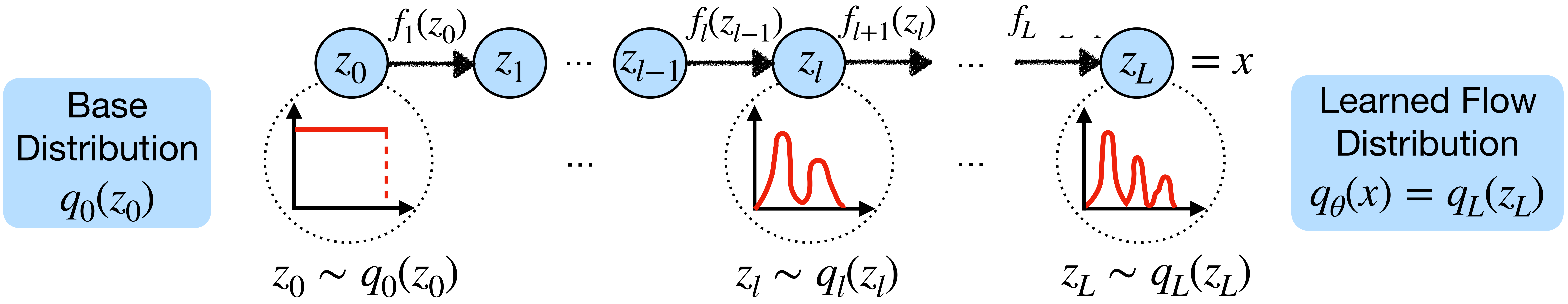
How to construct a Normalizing Flow?

Normalizing Flow



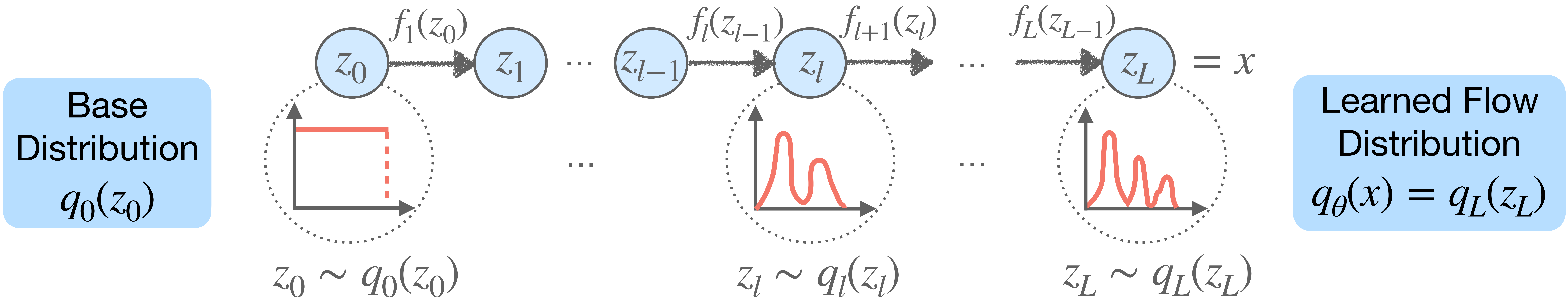
How to construct a Normalizing Flow?

Normalizing Flow = Sequence of invertible transformations:



How to sample from a Normalizing Flow?

Normalizing Flow = Sequence of invertible transformations:

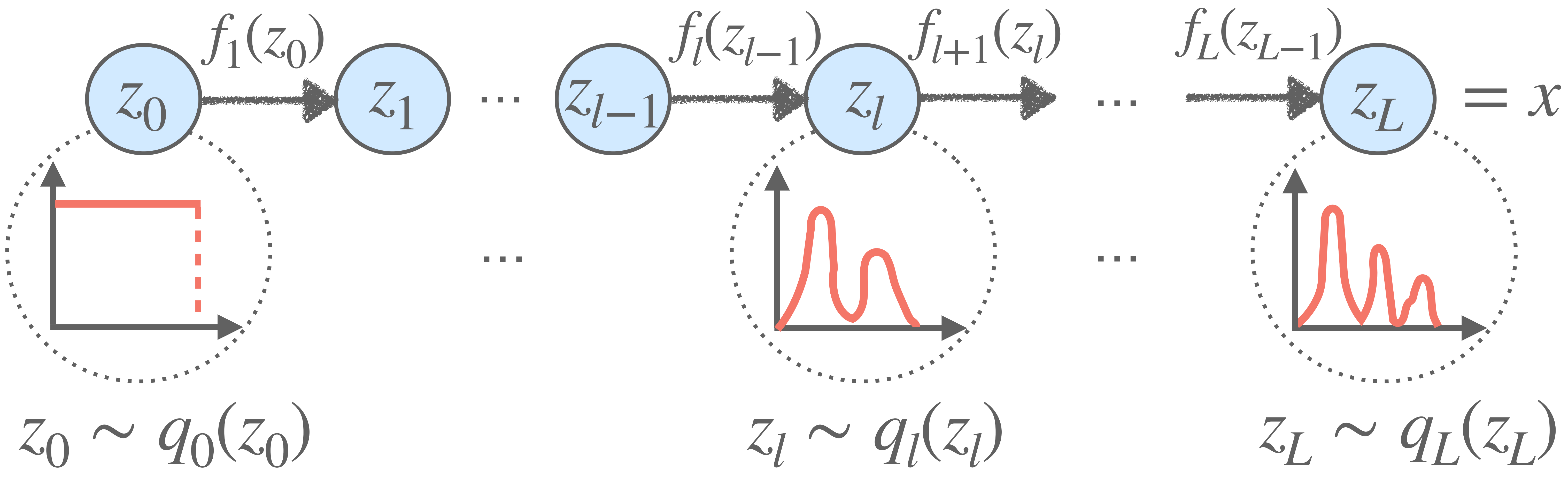


Sampling: $z_0 \sim q_0(z_0) \longrightarrow x = z_L = f_L \circ \ldots \circ f_1(z_0)$

How to evaluate the density?

Normalizing Flow = Sequence of invertible transformations:

Base Distribution $q_0(z_0)$



Learned Flow Distribution $q_\theta(x) = q_L(z_L)$

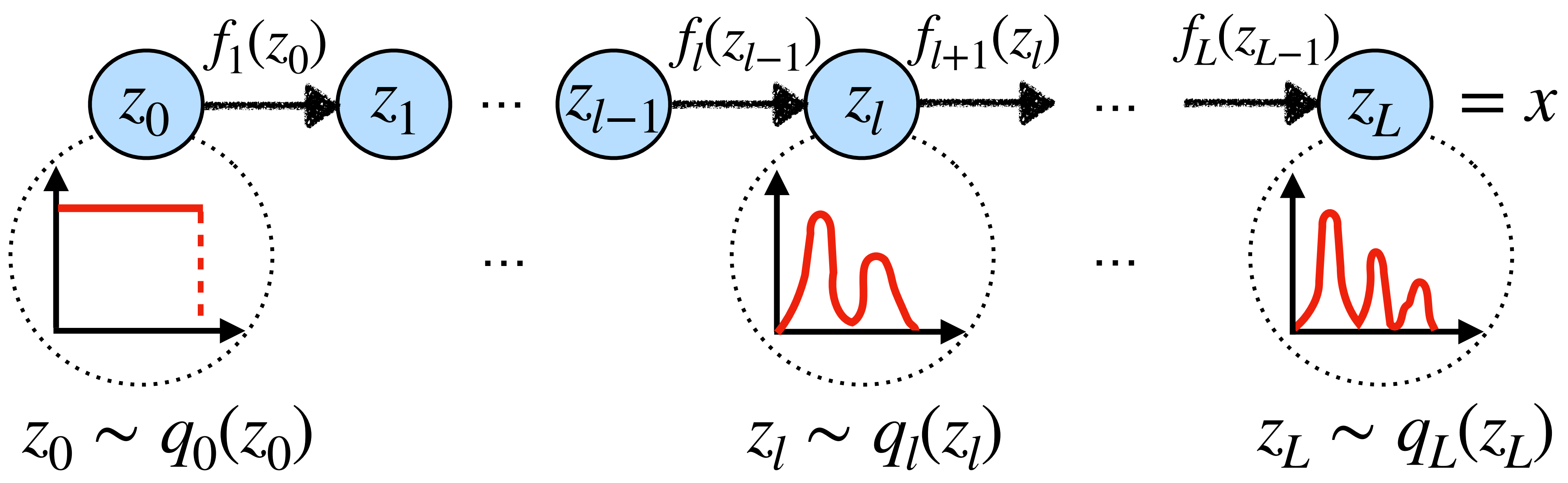
Evaluating the density: $z_0 = f_1^{-1} \circ \ldots \circ f_L^{-1}(x) \longleftarrow x = z_L = f_L(z_{L-1})$

$$q(x) = q_0\big(f^{-1}(x)\big) \left| \det \left(\frac{\partial f^{-1}(x)}{\partial x} \right) \right| = q_0(f^{-1}(x)) \left| \det J_{f^{-1}} \right|$$

Example: Gaussian with two modes

Base Distribution

$q_0(z_0)$



Learned Flow Distribution

$q_\theta(x) = q_L(z_L)$

Example:

Samples from target $p(x)$

$z_0 \sim q_0(z_0)$

$z_5 \sim q_5(z_5)$

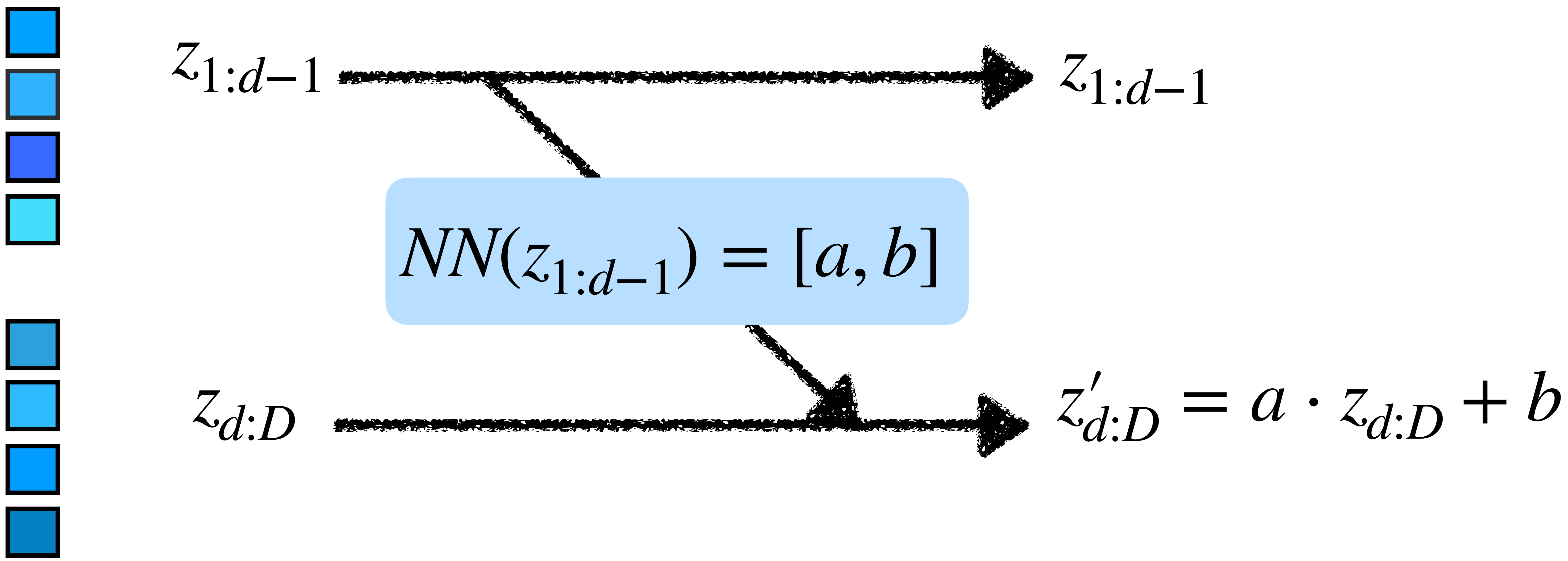
$q(x) = q_5(z_5)$

How to construct the transformations?

How to construct **invertible** transformations with **tractable** Jacobian:

- Linear transformation: $z_i = a \cdot z_{i-1} + b \rightarrow$ learn a and b by NN

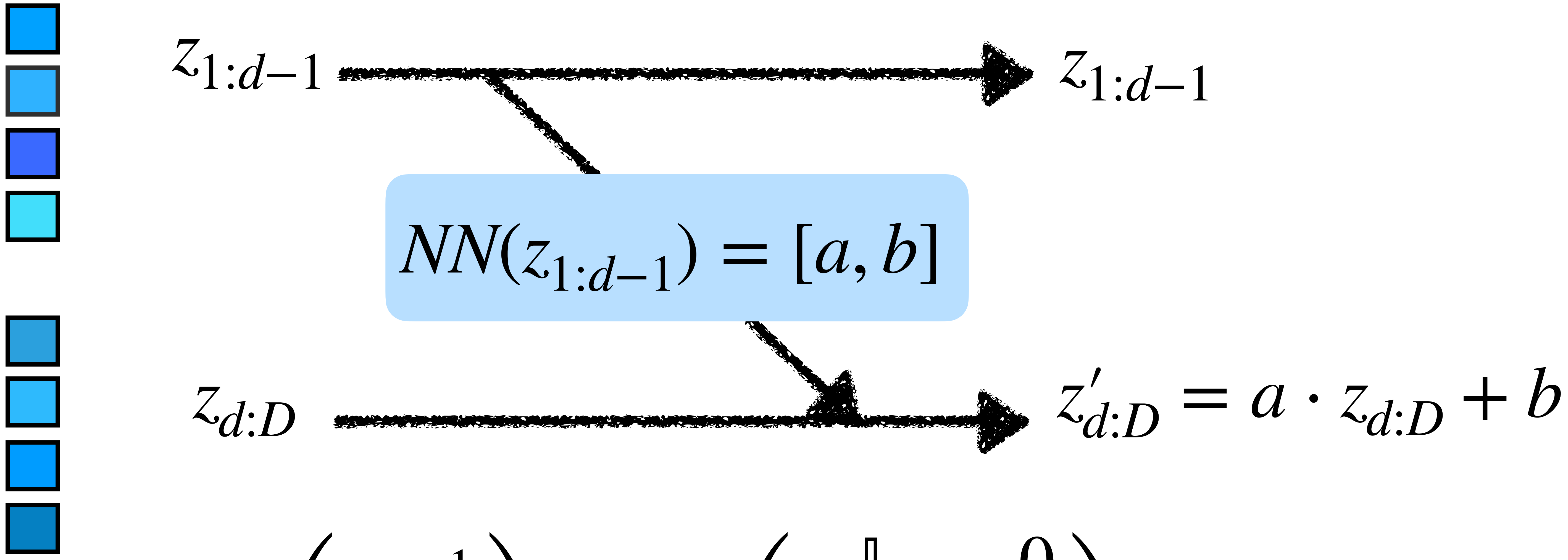
- Coupling layers: split $\vec{z} = (z_{1:d-1}, z_{d:D}) \in \mathbb{R}^D$



How to construct the transformations?

How to construct **invertible** transformations with **tractable** Jacobian:

- Linear transformation: $z_i = a \cdot z_{i-1} + b \rightarrow$ learn a and b by NN
- Coupling layers: split $\vec{z} = (z_{1:d-1}, z_{d:D}) \in \mathbb{R}^D$

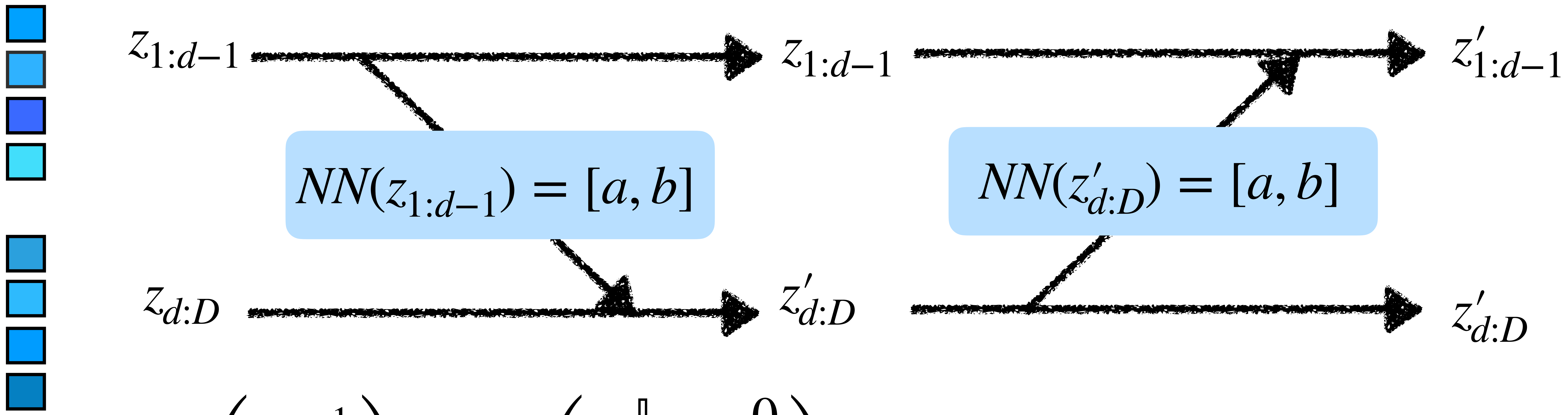


$$\det\left(\frac{\partial f^{-1}}{\partial z}\right) = \det\begin{pmatrix} \mathbb{I} & 0 \\ \frac{\partial z'_{d:D}}{\partial z_{1:d-1}} & \mathbb{I}a \end{pmatrix} = a^{D-d}$$

How to construct the transformations?

How to construct **invertible** transformations with **tractable** Jacobian:

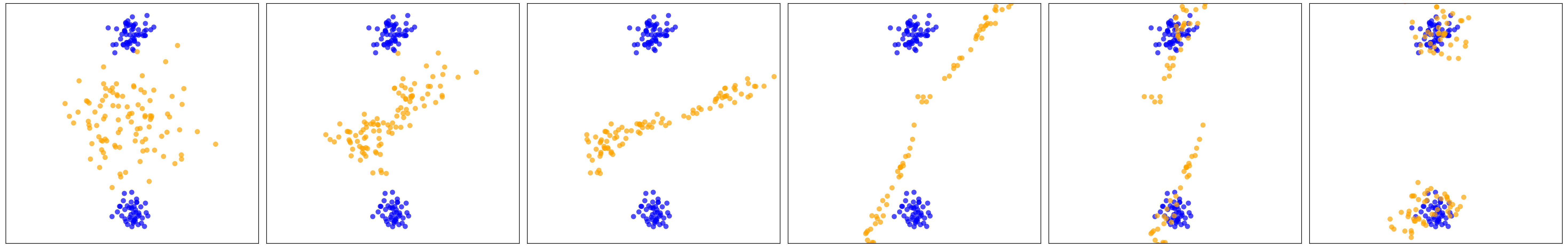
- Linear transformation: $z_i = a \cdot z_{i-1} + b \rightarrow$ learn a and b by NN
- Coupling layers: split $\vec{z} = (z_{1:d-1}, z_{d:D}) \in \mathbb{R}^D$

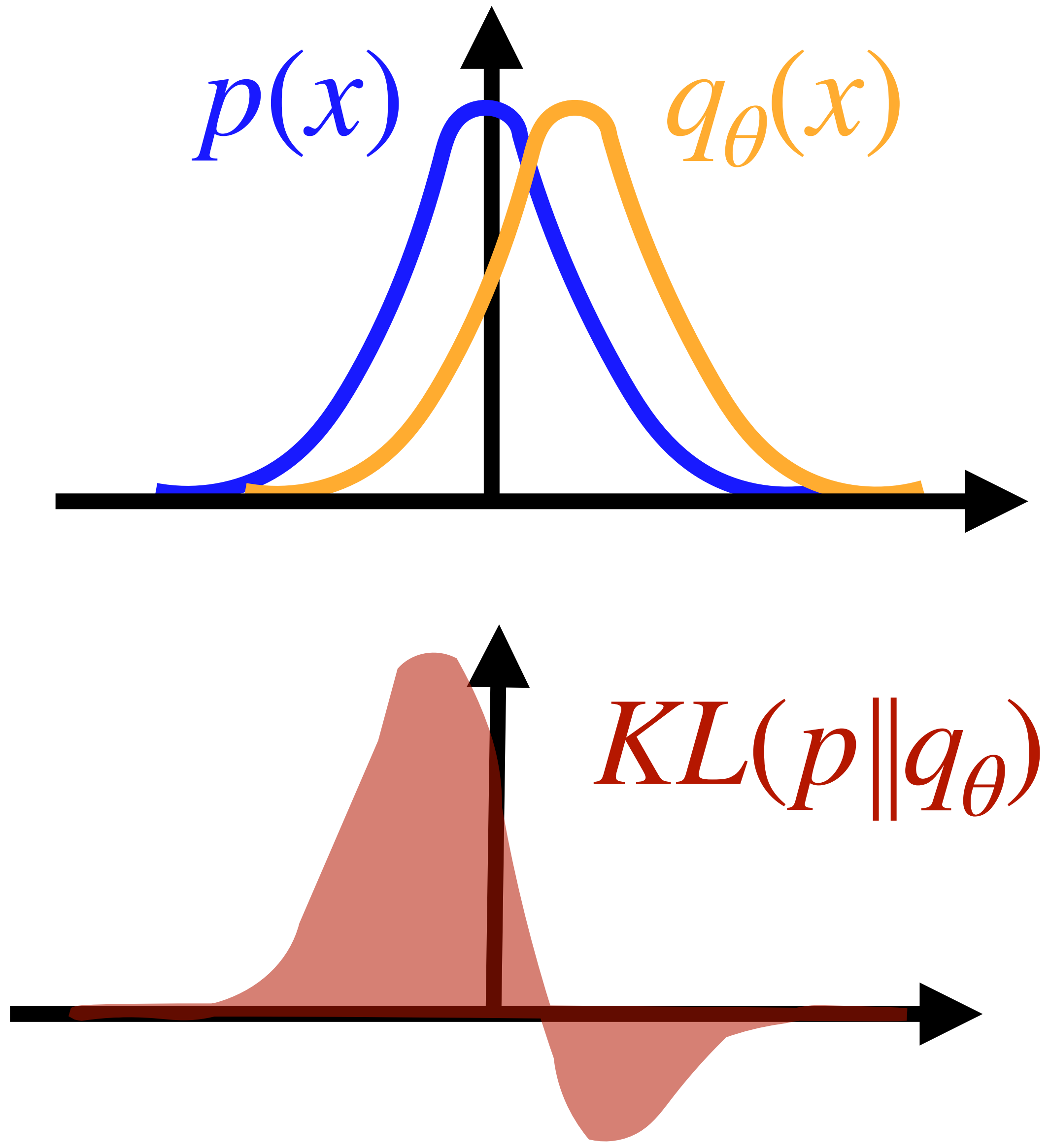


$$\det\left(\frac{\partial f^{-1}}{\partial z}\right) = \det\begin{pmatrix} \mathbb{1} & 0 \\ \frac{\partial z'_{d:D}}{\partial z_{1:d-1}} & \mathbb{1}a \end{pmatrix} = a^{D-d}$$

How to train a normalizing flow?

How can we compare two distributions?





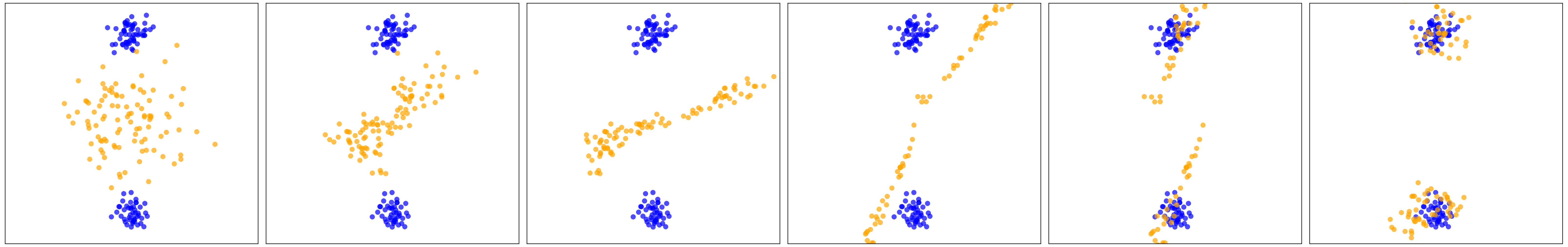
Kullback-Leibler divergence

$$\mathscr{L} = \mathrm{KL}(p \,\|\, q_\theta) = \mathbb{E}_{x \sim p(x)}\left[\log \frac{p(x)}{q_\theta(x)}\right] = -\sum_{i=1}^{N} \log q_\theta(x_i) + \text{const.}$$

→ minimize negative log-likelihood by evaluating the density of training samples

Normalizing flows: 3 Take-Aways

1. Normalizing flow = sequence of invertible transformations



2. Transformation constructed to have tractable Jacobian
   → Exact density evaluation, but limited architecture

3. Trained by evaluating and minimizing the negative log-likelihood of training data samples

How to normalize? - Three Model Families

1. Approximate Z_θ
   - Energy-based models

X Inaccurate probability

2. Use restricted NN architectures
   - Variational autoencoders
   - Autoregressive models
   - **Normalizing Flows** ✓

X Limited flexibility of the model

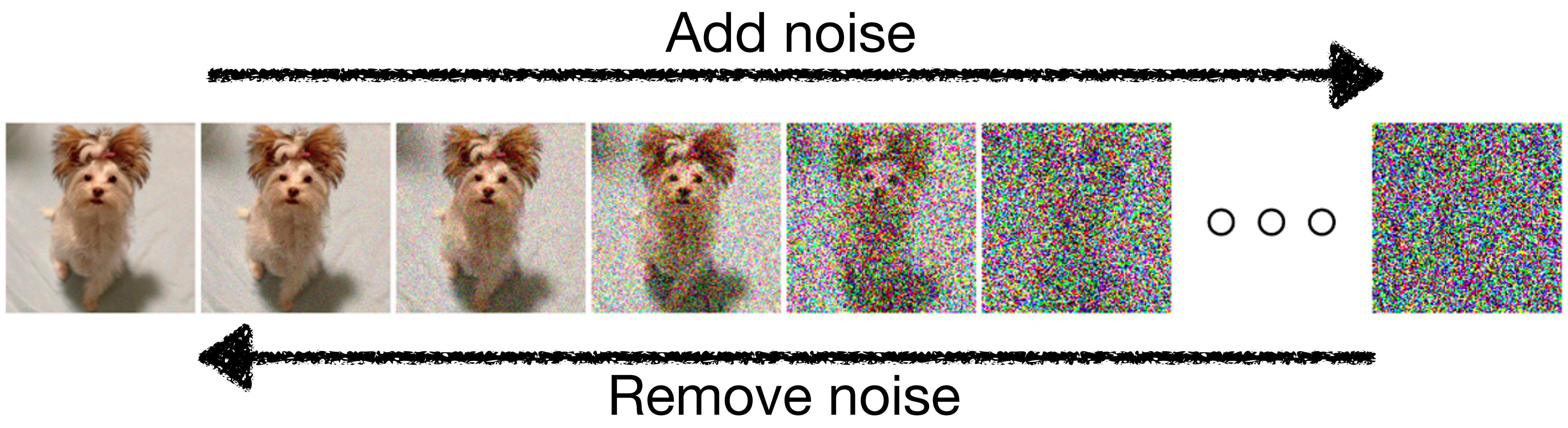
3. Model the generative process only
   - GANs
   - **Diffusion models**

X No/approximate evaluation of the probability

Diffusion Models

What is a diffusion model?

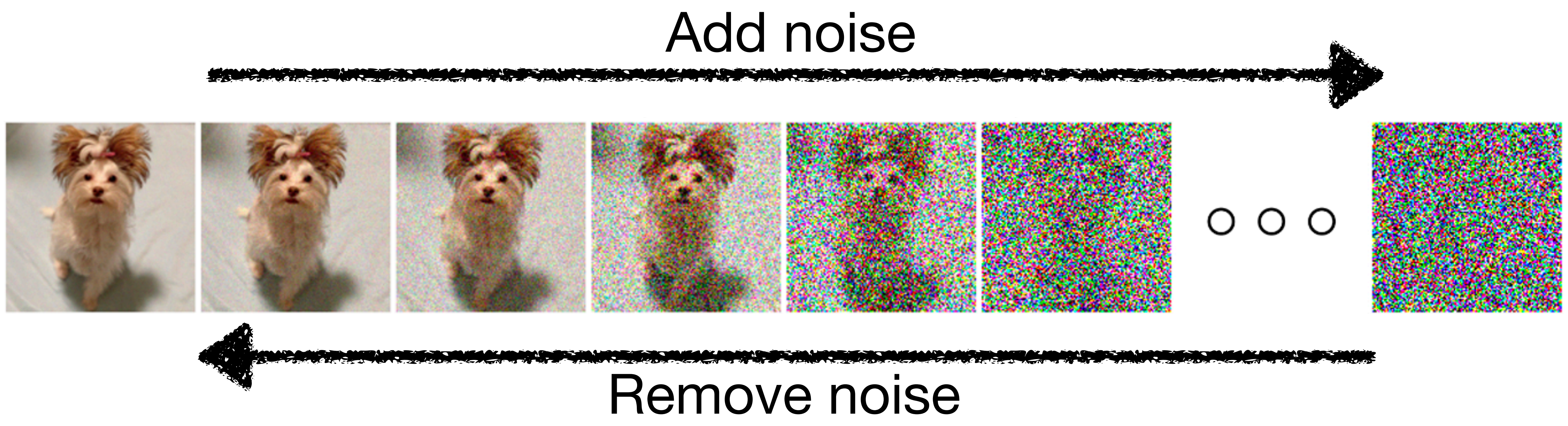
- Characterized by
  - progressively adding noise to the data
  - learning the reverse process



- Members of the diffusion family:
  - Denoising Diffusion Probabilistic Model
  - Score-Based Generative Models
  - Stochastic Differential Equations

What is a diffusion model?

- Characterized by
  - progressively adding noise to the data
  - learning the reverse process



- Members of the diffusion family:
  - **Denoising Diffusion Probabilistic Model**
  - Score-Based Generative Models
  - Stochastic Differential Equations

How does a diffusion model work?

- Forward process: add Gaussian noise to data

$p(x_0)$

$\hookrightarrow p(x_1|x_0) = \mathcal{N}(x_1 \,|\, \sqrt{\alpha_1} x_0, (1-\alpha_1)\mathbb{I})$

$\hookrightarrow p(x_2|x_1, x_0) = \mathcal{N}(x_2 \,|\, \sqrt{\alpha_2} x_1, (1-\alpha_2)\mathbb{I})$

$\hookrightarrow \dots$

- Why Gaussians?

$$p(x_t|x_0) = \mathcal{N}\left(x_t \,\middle|\, \sqrt{\prod_{s=0}^{t} \alpha_s}\; x_0,\; \left(1 - \prod_{s=0}^{t} \alpha_s\right)\mathbb{I}\right) \equiv \mathcal{N}\left(x_t \middle| \sqrt{\bar{\alpha}_t}\, x_0,\; (1 - \sqrt{\bar{\alpha}_t})\mathbb{I}\right)$$

How does a diffusion model work?

- Forward process: add Gaussian noise to data

- Why Gaussians?
  → reparametrization trick: sample noise $\epsilon \sim \mathcal{N}(\epsilon \,|\, 0, \mathbb{I})$

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + (1 - \sqrt{\bar{\alpha}_t})\, \epsilon$$

How does a diffusion model work?

- Reverse process: “Remove” noise → learn by DNN

x_0 $\dots$ x_{t-1} x_t x_T

- How to remove noise?

$$x_t \longrightarrow \boxed{NN(x_t, t) = [\mu_\theta, \Sigma_\theta]}$$

$$\downarrow$$

$$q_\theta(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}|\,\mu_\theta(x_t,t),\ \Sigma_\theta(x_t,t)\right) \longrightarrow x_{t-1} \sim q_\theta(x_{t-1}|x_t)$$

How to sample from a diffusion model?

- Sample random noise

$x_0 \quad x_1 \quad \dots \quad x_{t-1} \quad x_t \quad x_T$

$$x_T \sim \mathcal{N}(x_T \,|\, 0, \mathbb{I})$$

- Go through reverse process until $t = 0$:

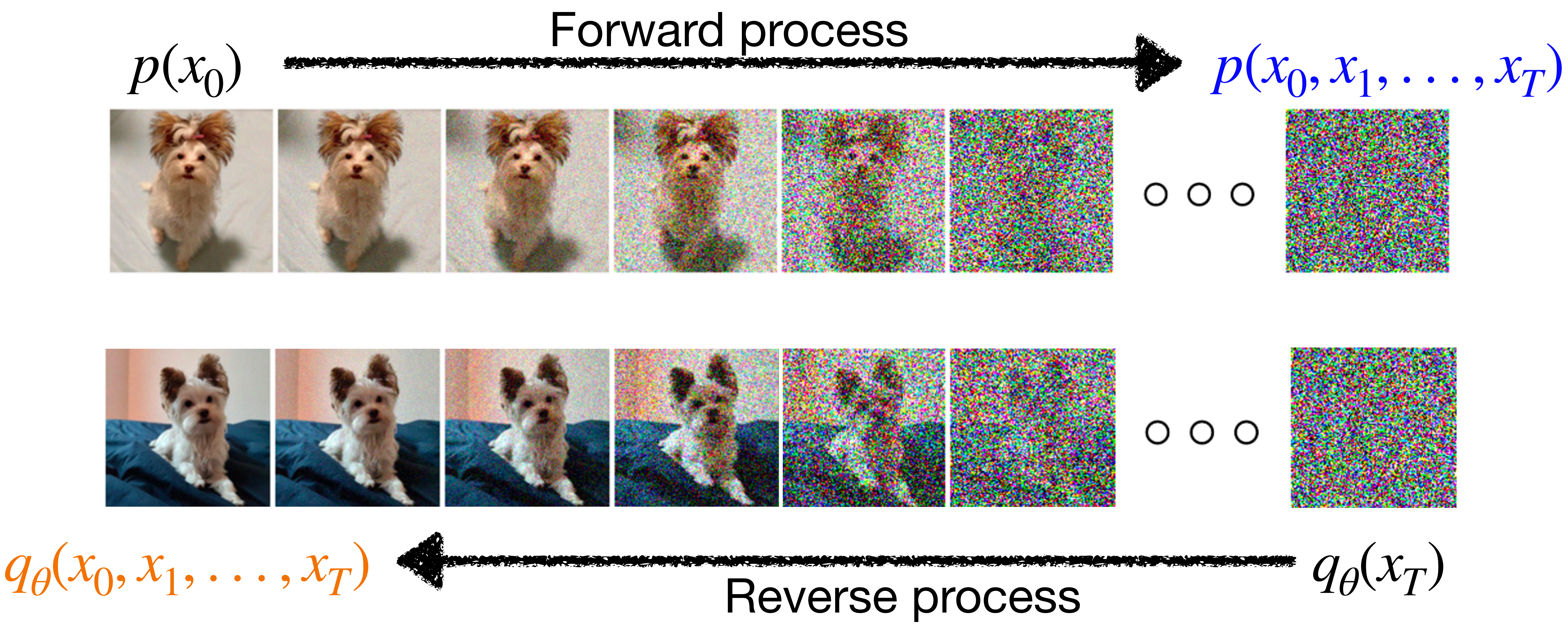
$$x_1 \longrightarrow \boxed{NN(x_1, t) = [\mu_\theta, \Sigma_\theta]}$$

$$\downarrow$$

$$q_\theta(x_0|x_1) = \mathcal{N}\left(x_0 |\, \mu_\theta(x_1, t),\ \Sigma_\theta(x_1, t)\right) \longrightarrow x_0 \sim q_\theta(x_0|x_1)$$

How to train a diffusion model?

- Compare joint distributions with Kullback-Leibler divergence:



$$\mathscr{L} = KL\Big(p(x_0, x_1, \ldots, x_T) \| q_\theta(x_0, \ldots, x_T)\Big) = -\mathbb{E}_{p(x_0, x_1, \ldots, x_T)}\Big[\log q_\theta(x_0, x_1, \ldots, x_T)\Big] + \text{const.}$$

Why is the density evaluation not exact?

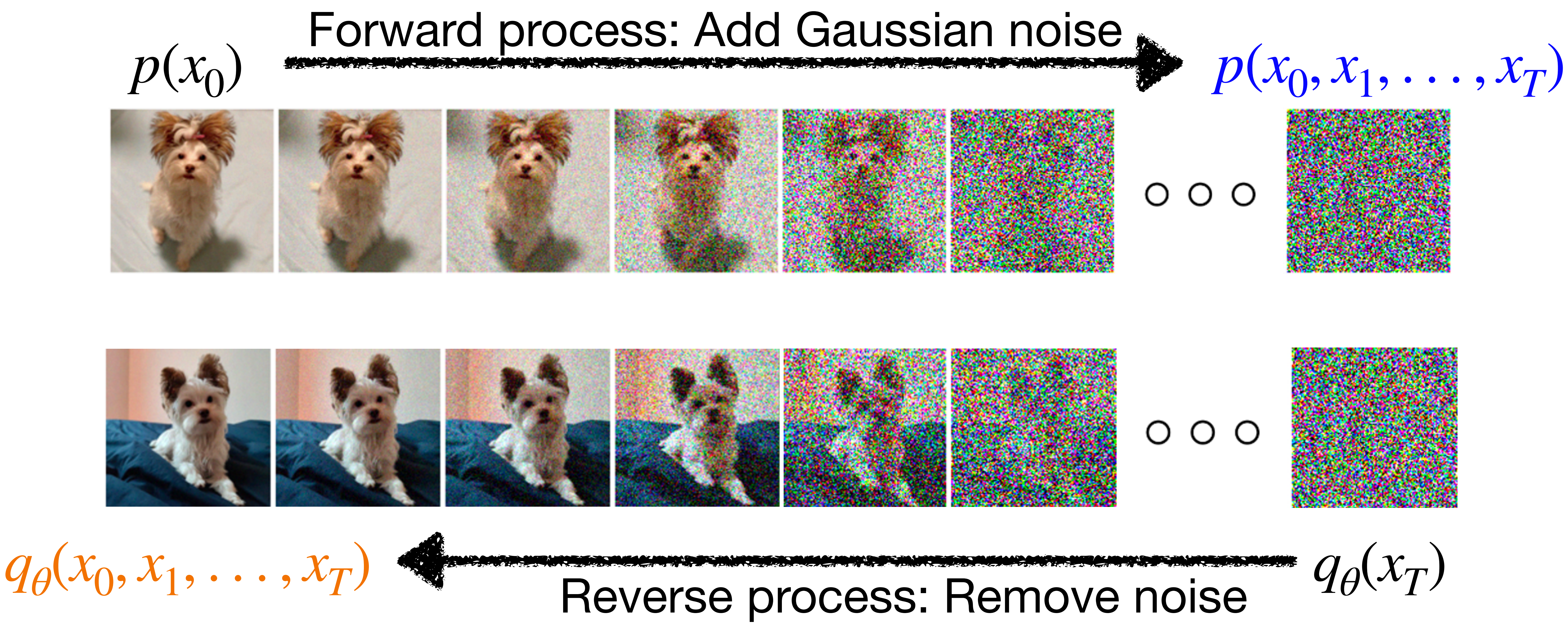
- In overview: Diffusion Models

X No/approximate evaluation of the probability

- Density evaluation only exact for $T \to \infty$
  $\to q_\theta(x_0, x_1, \ldots, x_T)$ for e.g. $T = 10$ is only approximation!

Diffusion Models: 3 Take-Aways

1. Diffusion Model = Model that learns how to remove noise



2. Parametrize mean and variance of Gaussian by DNN

3. Density Evaluation is approximation

How to normalize? - Three Model Families

1. Approximate Z_θ
   - Energy-based models

X Inaccurate probability

2. Use restricted NN architectures
   - Variational autoencoders
   - Autoregressive models
   - **Normalizing Flows** ✓

X Limited flexibility of the model

3. Model the generative process only
   - GANs
   - **Diffusion models** ✓
   - Score matching

X No/approximate evaluation of the probability

Recommended References for the Interested

- Energy-based models:
  - Song and Kingma, "How to train your Energy-Based Model", 2021
- Normalizing Flows:
  - Rezende and Mohamed, "Variational Inference with Normalizing Flows", 2016
  - Kobyzev, et al., "Normalizing Flows: An Introduction and Review of Current Methods", 2021
- Diffusion Models:
  - Yang, et al. "Diffusion Models: A Comprehensive Survey of Methods and Applications", 2024, under review
- Score-Matching:
  - Song, et al., "Score-Based Generative Modeling through Stochastic Differential Equations", 2021
- Talk by Yang Song on "Diffusion and Score-Based Generative Models", 2022

Thank you!
Do you have any questions?