Learning normalized probability models with dual score matching

Center for Data Science NYU

Florentin Guth
New York University
& Flatiron Institute

FLATIRON INSTITUTE
Center for Computational Neuroscience

Zahra Kadkhodaie
Flatiron Institute

Eero Simoncelli
NYU & Flatiron Institute

Probabilistic modeling from samples

Dataset

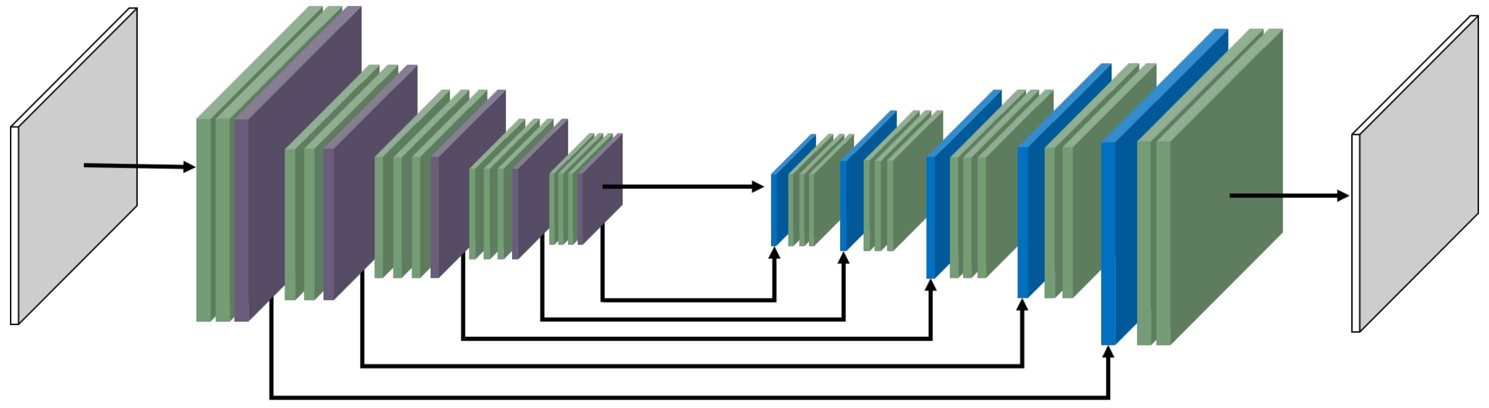
$$x_1, \ldots, x_n \sim p(x)$$

(Hinton and Sejnowski, 1986; Hinton, 2002; Teh et al., 2003; Bengio et al., 2003; LeCun and Huang, 2005; Gutmann and Hyvärinen, 2010; …)

Probabilistic modeling from samples

Dataset

$$x_1, \ldots, x_n \sim p(x)$$



Learning

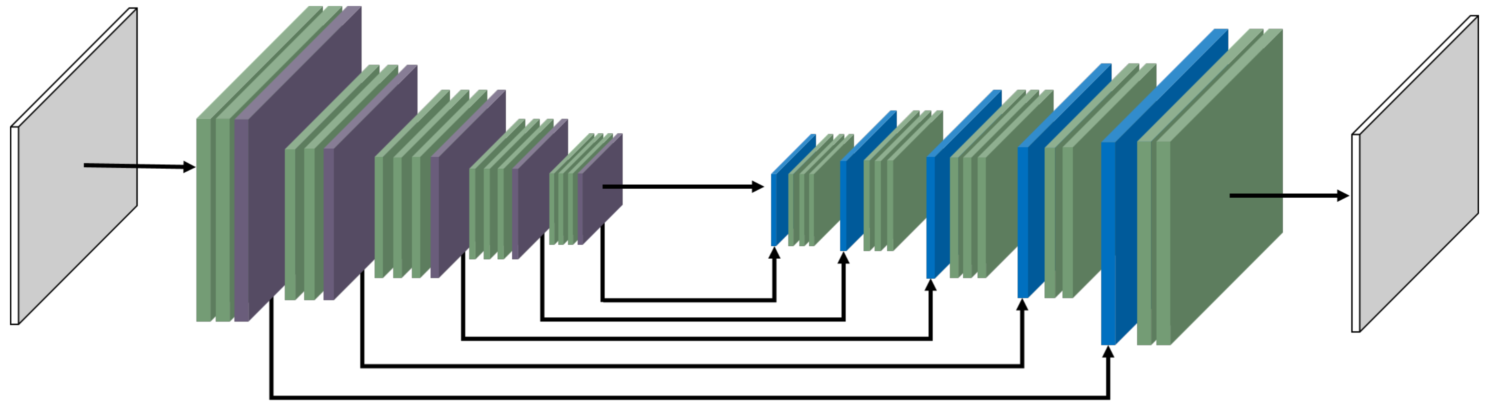
(deep network)

(Hinton and Sejnowski, 1986; Hinton, 2002; Teh et al., 2003; Bengio et al., 2003; LeCun and Huang, 2005; Gutmann and Hyvärinen, 2010; …)

Probabilistic modeling from samples

Dataset

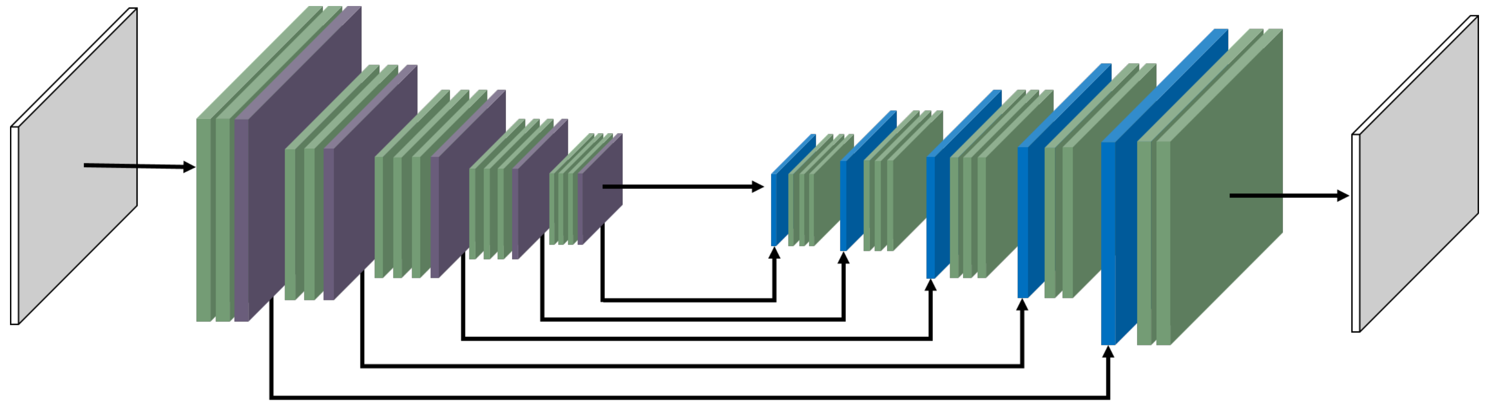
$x_1, \ldots, x_n \sim p(x)$ → **Learning** → Density model $p_\theta(x)$



(deep network)

(Hinton and Sejnowski, 1986; Hinton, 2002; Teh et al., 2003; Bengio et al., 2003; LeCun and Huang, 2005; Gutmann and Hyvärinen, 2010; …)

Probabilistic modeling from samples



Dataset

$x_1, \ldots, x_n \sim p(x)$ → **Learning** (deep network) → Density model $p_\theta(x)$

- How do we learn it?
  - Can we trust it?
- What can we use it for?
  - What does it tell us about the data?

(Hinton and Sejnowski, 1986; Hinton, 2002; Teh et al., 2003; Bengio et al., 2003; LeCun and Huang, 2005; Gutmann and Hyvärinen, 2010; …)

Are deep generative models memorizing?

Definitely if they're training on a single image!

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

Definitely if they're training on a single image!

Training set size $n = 1$

Generated image

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

Definitely if they're training on a single image!

Training set size $n = 1$

Generated image

Closest training image

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

Definitely if they’re training on a single image!

Training set size $n = 1$

Generated image

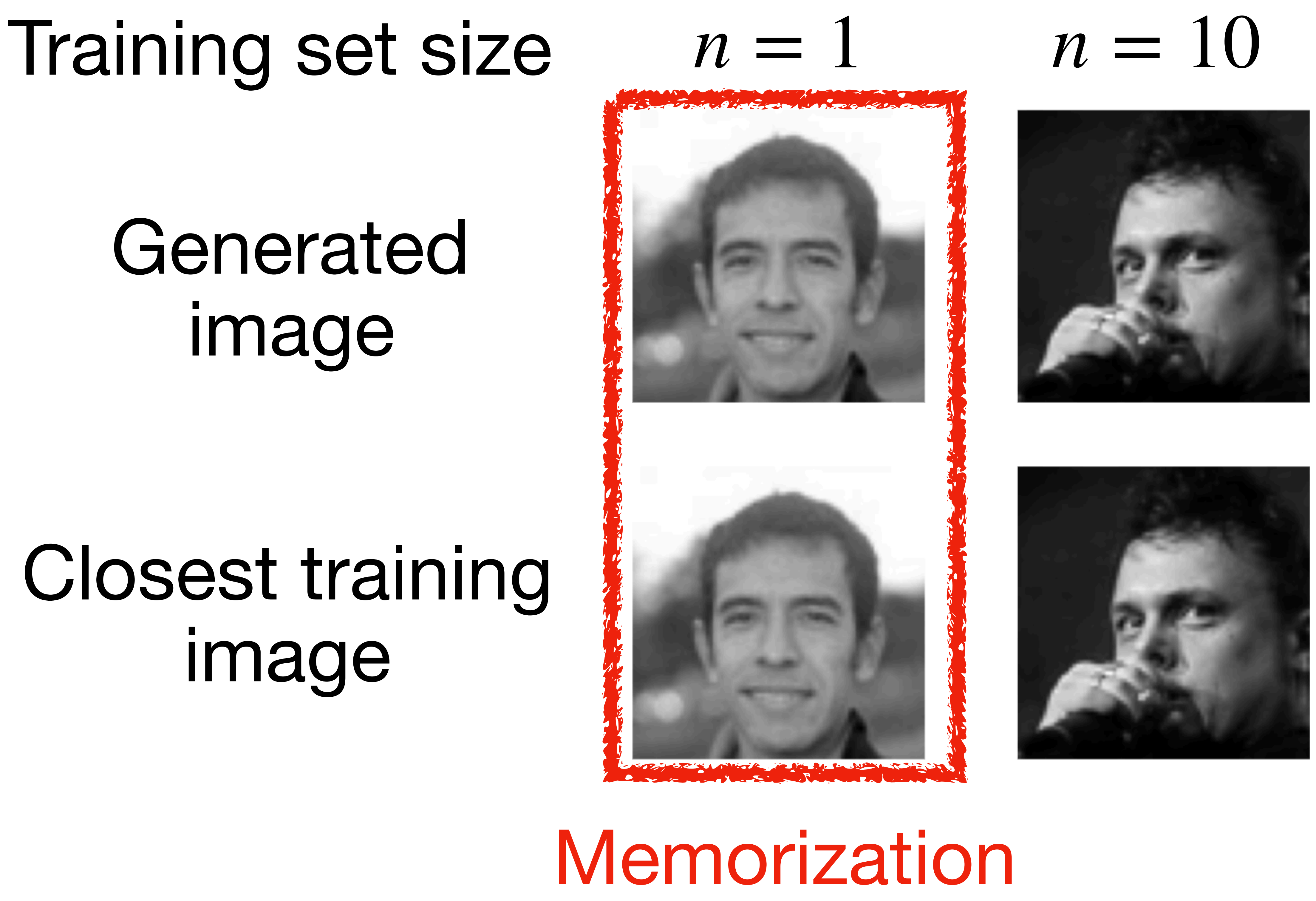
Closest training image

Memorization

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

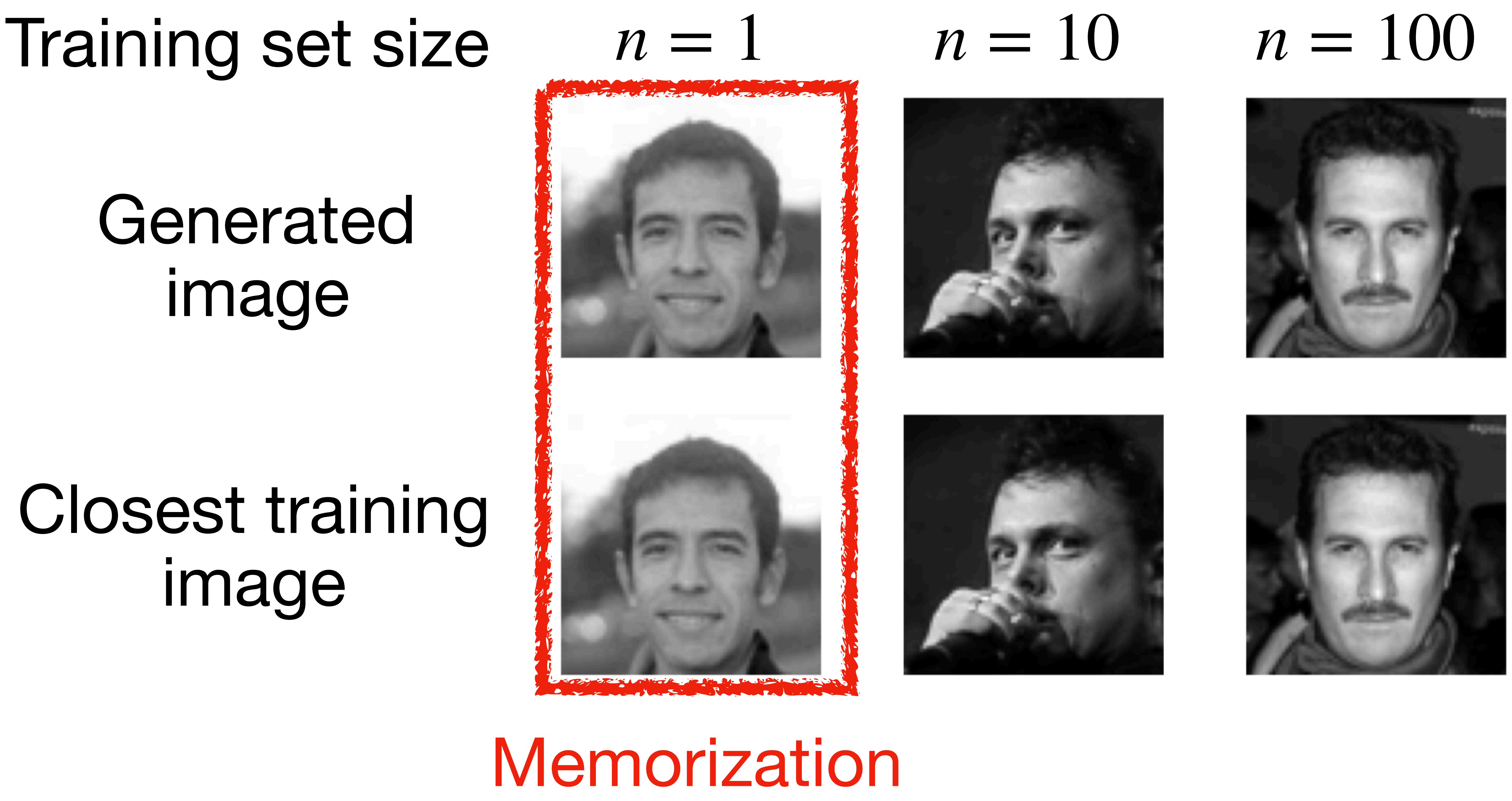
Definitely if they're training on a single image!



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

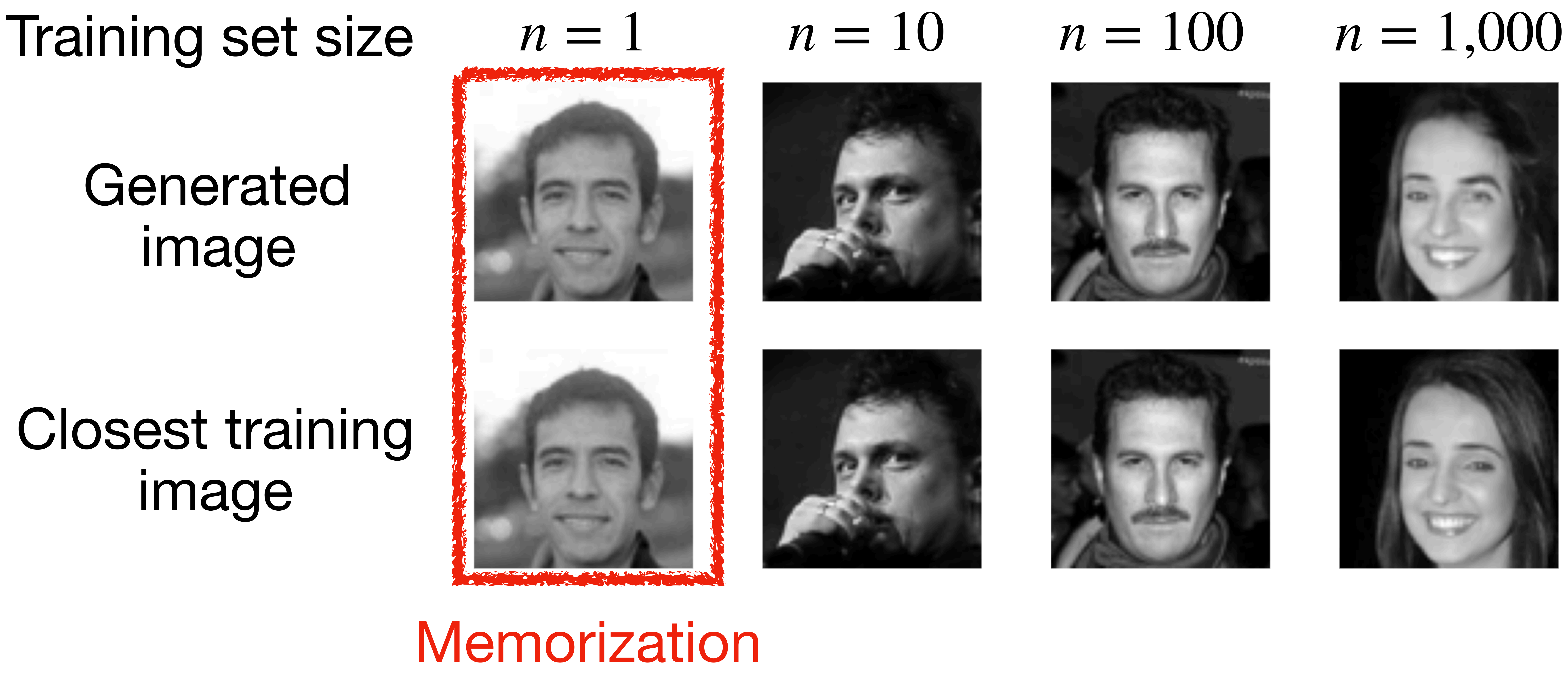
Definitely if they're training on a single image!



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

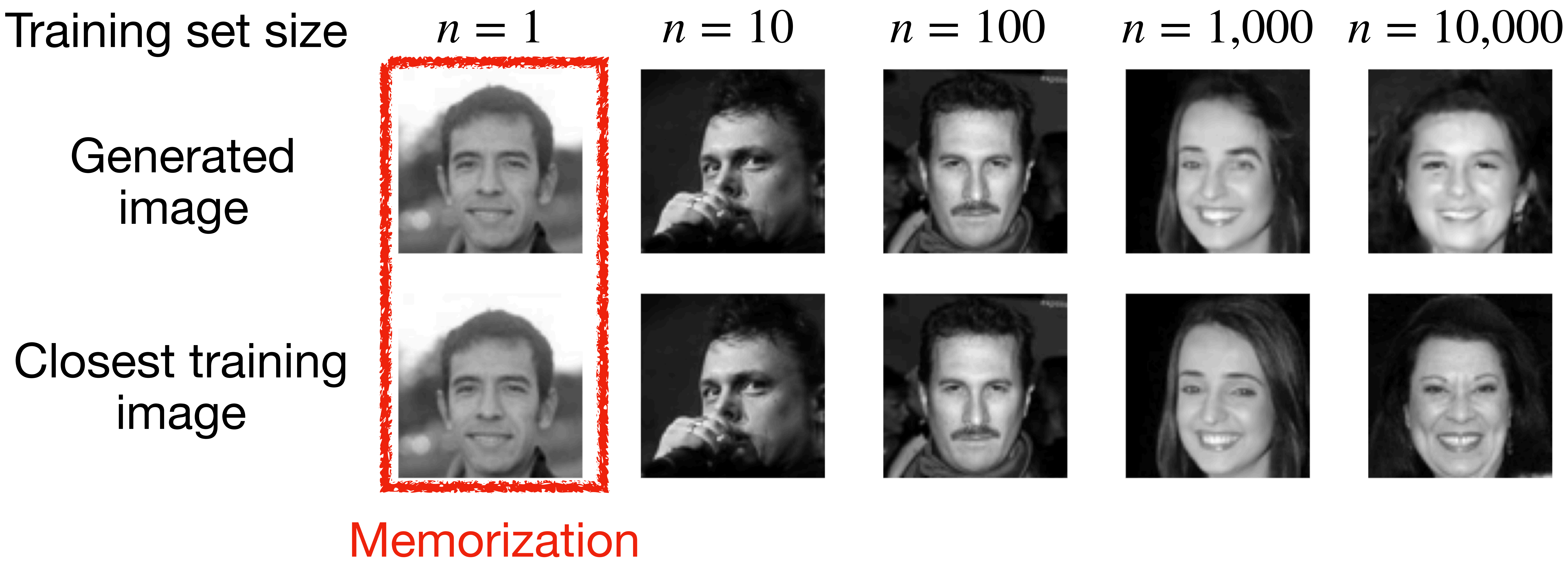
Definitely if they're training on a single image!



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

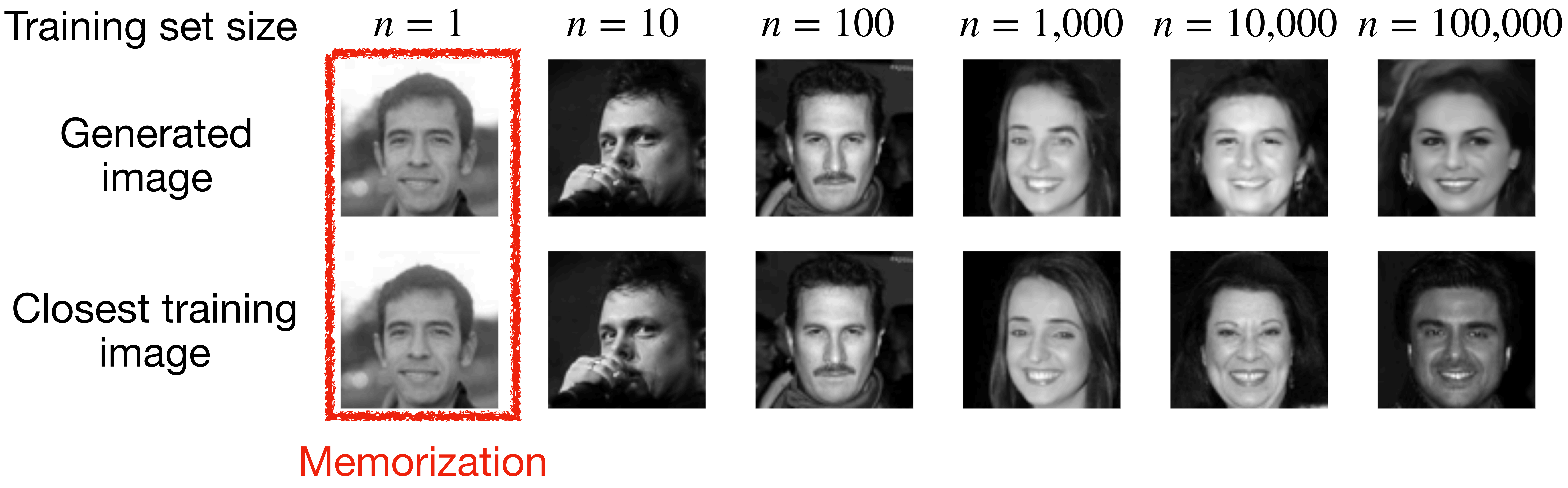
Definitely if they're training on a single image!



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

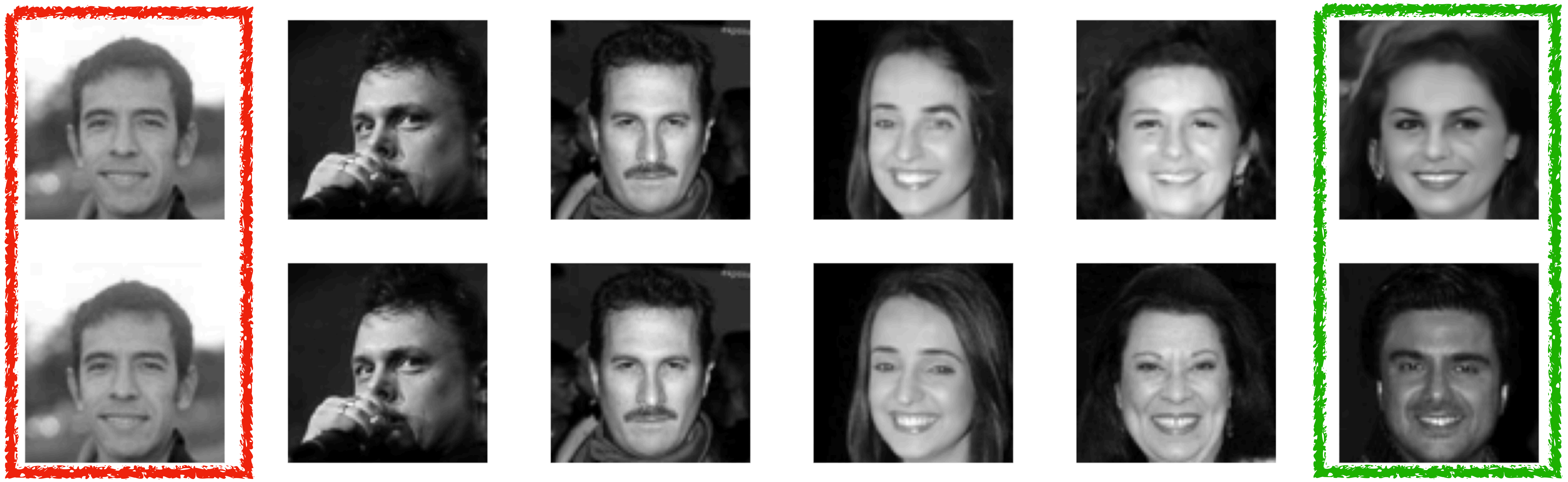
Definitely if they're training on a single image!



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

Definitely if they're training on a single image!



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

Training set size $n = 1$ $n = 10$ $n = 100$ $n = 1{,}000$ $n = 10{,}000$ $n = 100{,}000$

Generated image

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images

Training set size	$n = 1$	$n = 10$	$n = 100$	$n = 1{,}000$	$n = 10{,}000$	$n = 100{,}000$

Generated image

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images

Training set size: $n = 1$, $n = 10$, $n = 100$, $n = 1{,}000$, $n = 10{,}000$, $n = 100{,}000$

Generated image (A)

Generated image (B)

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).

Training set size $n = 1$ $n = 10$ $n = 100$ $n = 1{,}000$ $n = 10{,}000$ $n = 100{,}000$

Generated image (A)

Generated image (B)

(Kadkhodaie et al., 2024)

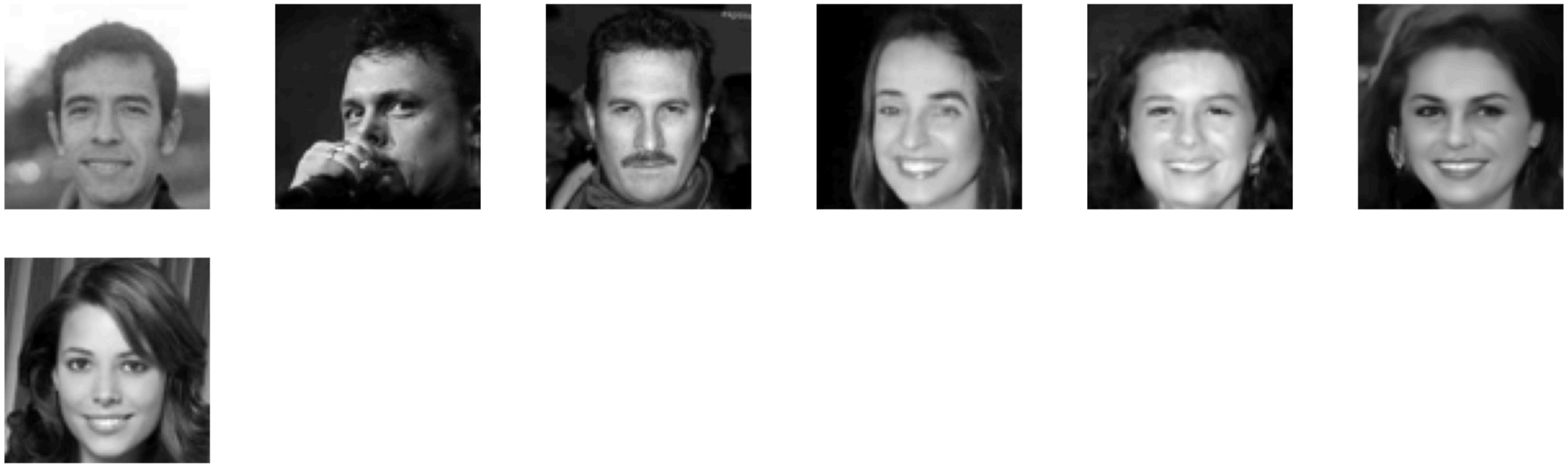
Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).

Training set size $n = 1$ $n = 10$ $n = 100$ $n = 1{,}000$ $n = 10{,}000$ $n = 100{,}000$

Generated image (A)

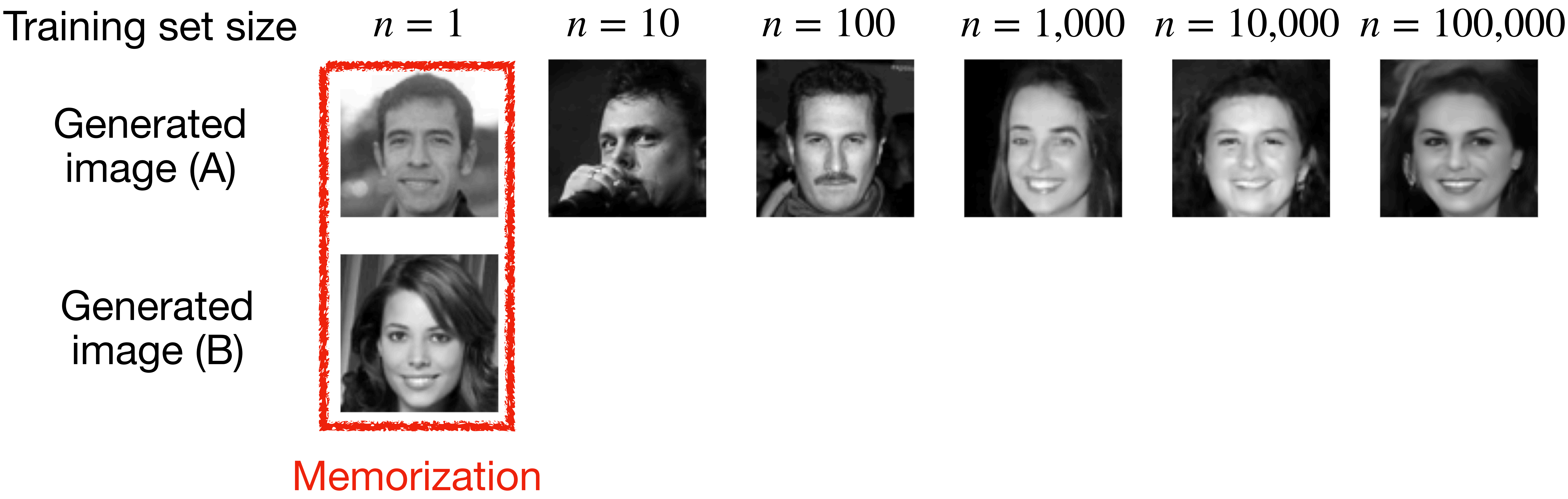
Generated image (B)



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

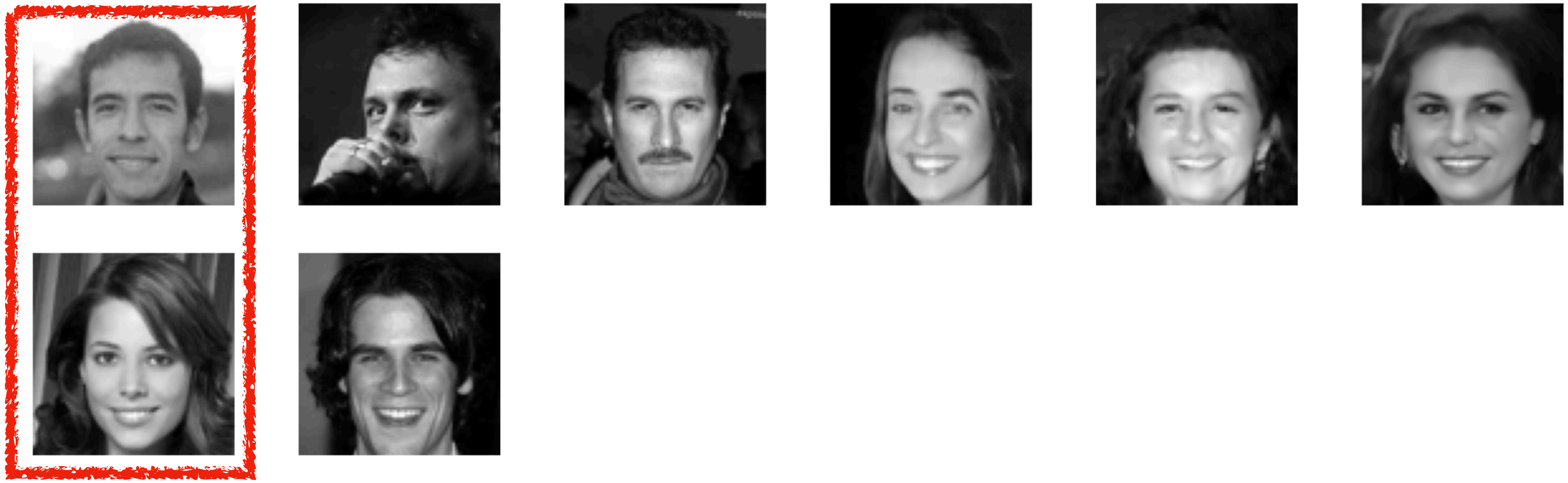
- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).



(Kadkhodaie et al., 2024)

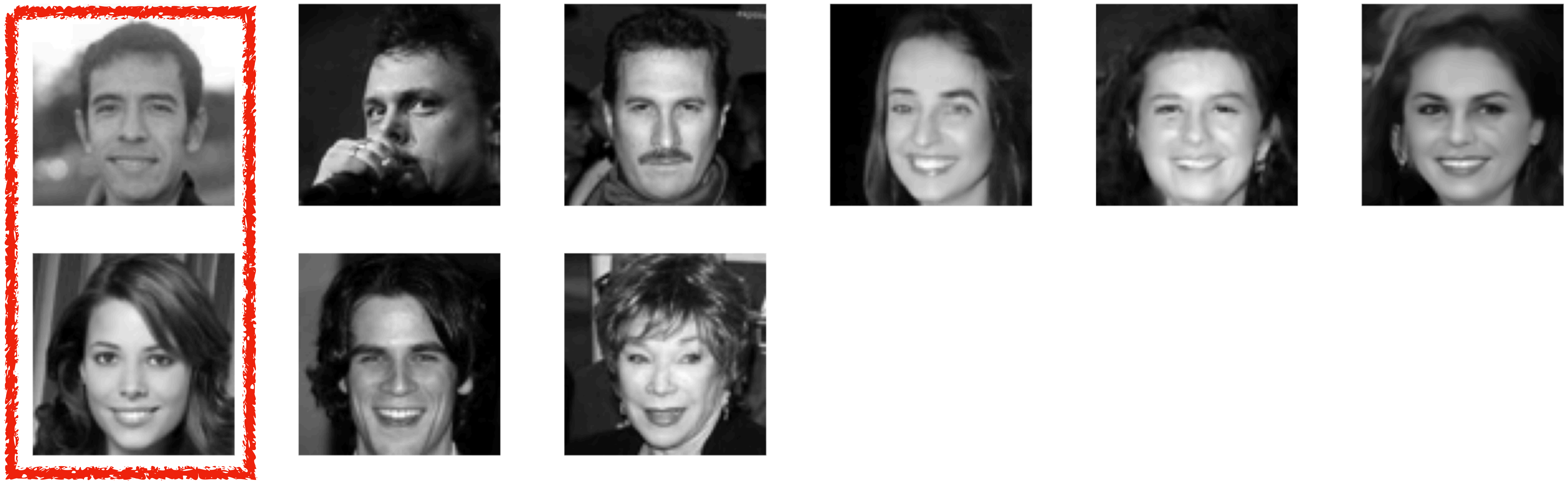
Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).

Training set size $n = 1$ $n = 10$ $n = 100$ $n = 1{,}000$ $n = 10{,}000$ $n = 100{,}000$

Generated image (A)

Generated image (B)

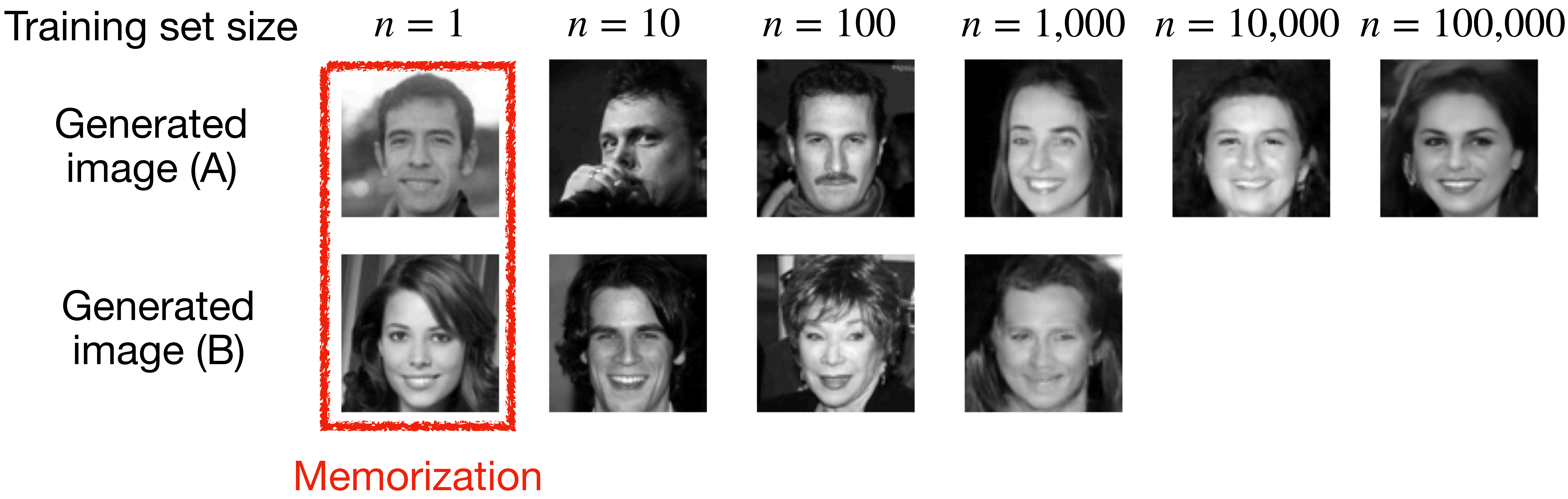


Memorization

(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

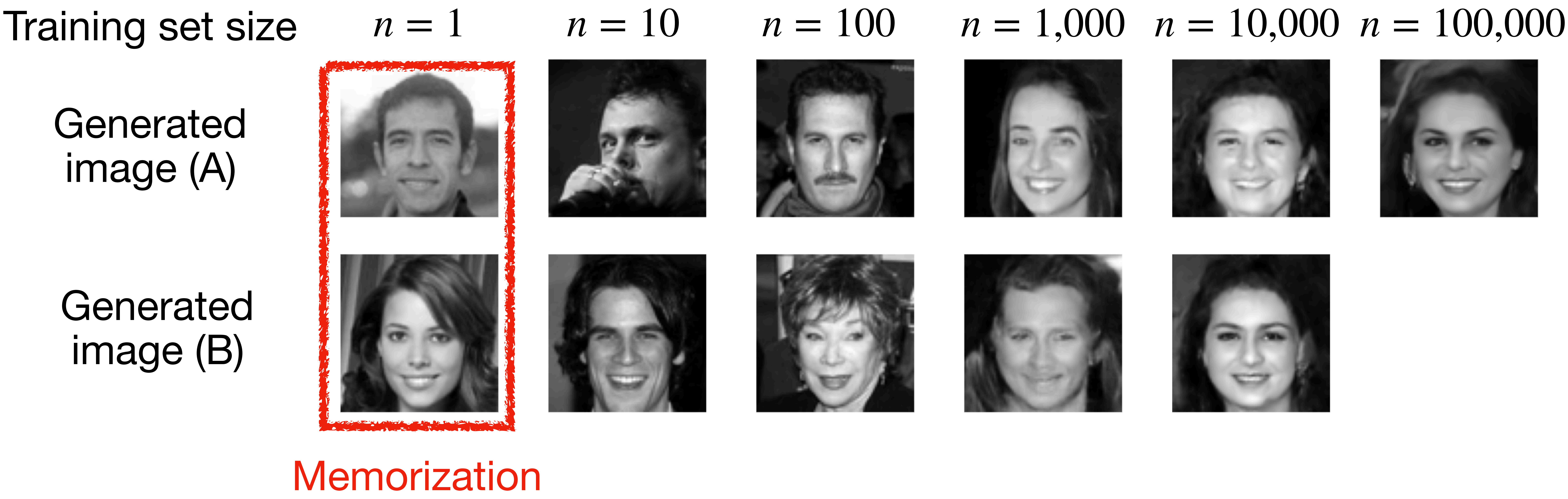
- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

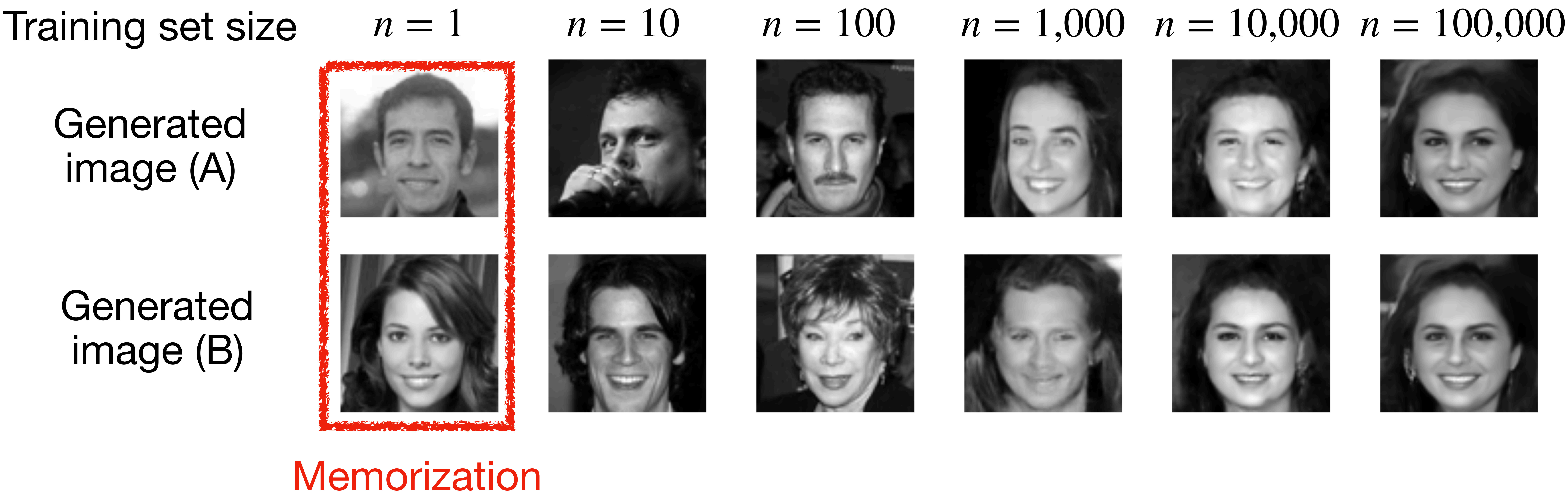
- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).



(Kadkhodaie et al., 2024)

Are deep generative models memorizing?

- Train another model on an another **non-overlapping** set of n images
- Generate samples **from the same noise sample** (and injected noise).

Training set size $n = 1$ $n = 10$ $n = 100$ $n = 1{,}000$ $n = 10{,}000$ $n = 100{,}000$

Generated image (A)

Generated image (B)

Memorization

Generalization!

(Kadkhodaie et al., 2024)

It should be hard to learn a probabilistic model!

Curse of dimensionality:

It should be hard to learn a probabilistic model!

Curse of dimensionality:

- **Statistical** challenge: we only have $n \ll \exp(d)$ samples
  - We need very strong inductive biases (network architecture, …)

It should be hard to learn a probabilistic model!

Curse of dimensionality:

- **Statistical** challenge: we only have $n \ll \exp(d)$ samples
  - We need very strong inductive biases (network architecture, …)

- **Computational** challenge: we cannot compute normalizing constants

It should be hard to learn a probabilistic model!

Curse of dimensionality:

- **Statistical** challenge: we only have $n \ll \exp(d)$ samples
  - We need very strong inductive biases (network architecture, …)

- **Computational** challenge: we cannot compute normalizing constants

$$p_\theta(x) = \frac{1}{Z_\theta} \mathrm{e}^{-U_\theta(x)}$$

It should be hard to learn a probabilistic model!

Curse of dimensionality:

- **Statistical** challenge: we only have $n \ll \exp(d)$ samples
  - We need very strong inductive biases (network architecture, …)

- **Computational** challenge: we cannot compute normalizing constants

$$p_\theta(x) = \frac{1}{Z_\theta} \mathrm{e}^{-U_\theta(x)} \qquad Z_\theta = \int \mathrm{e}^{-U_\theta(x)} \mathrm{d}x$$

It should be hard to learn a probabilistic model!

Curse of dimensionality:

- **Statistical** challenge: we only have $n \ll \exp(d)$ samples
  - We need very strong inductive biases (network architecture, …)

- **Computational** challenge: we cannot compute normalizing constants

$$p_\theta(x) = \frac{1}{Z_\theta} \mathrm{e}^{-U_\theta(x)} \quad Z_\theta = \int \mathrm{e}^{-U_\theta(x)} \mathrm{d}x \quad -\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

It should be hard to learn a probabilistic model!

Curse of dimensionality:

- **Statistical** challenge: we only have $n \ll \exp(d)$ samples
  - We need very strong inductive biases (network architecture, …)

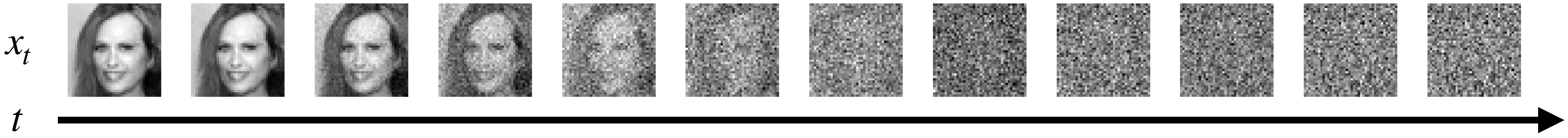
- **Computational** challenge: we cannot compute normalizing constants

$$p_\theta(x) = \frac{1}{Z_\theta} \mathrm{e}^{-U_\theta(x)} \quad Z_\theta = \int \mathrm{e}^{-U_\theta(x)} \mathrm{d}x \quad -\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

  - How come diffusion models solve this?

Diffusion models to the rescue

Introduce a diffusion process $(x_t)_{t \geq 0}$ and model the entire trajectories:



(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

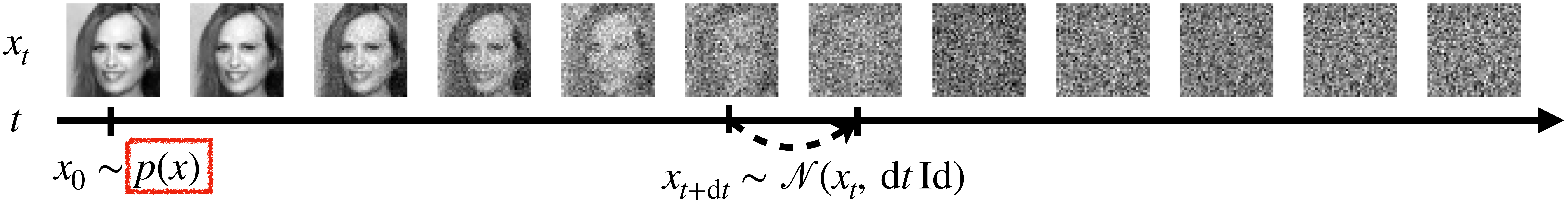
Diffusion models to the rescue

Introduce a diffusion process $(x_t)_{t \geq 0}$ and model the entire trajectories:

(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

Diffusion models to the rescue

Introduce a diffusion process $(x_t)_{t \geq 0}$ and model the entire trajectories:



(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

Diffusion models to the rescue

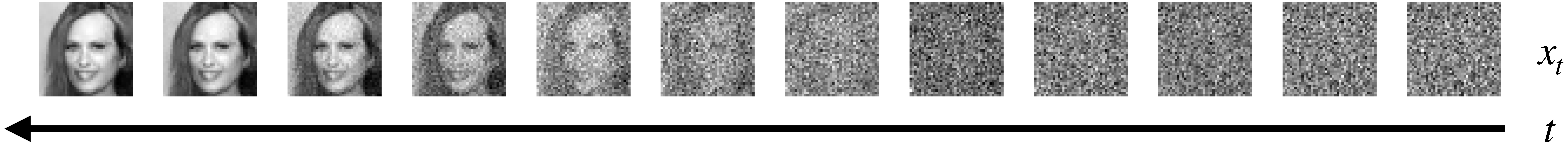
Introduce a diffusion process $(x_t)_{t \geq 0}$ and model the entire trajectories:

(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

Diffusion models to the rescue

Introduce a diffusion process $(x_t)_{t \geq 0}$ and model the entire trajectories:

With a time reversal, we can define the process just from **scores** rather than densities:



(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

Diffusion models to the rescue

Introduce a diffusion process $(x_t)_{t \geq 0}$ and model the entire trajectories:

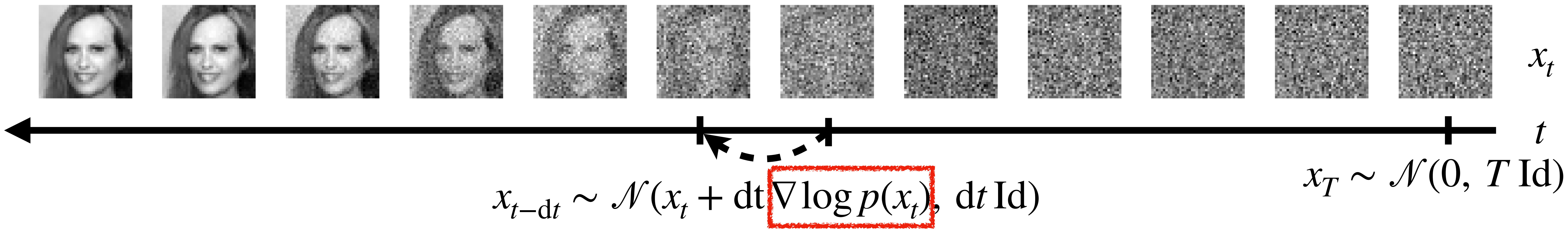
With a time reversal, we can define the process just from **scores** rather than densities:

(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

Diffusion models to the rescue

Introduce a diffusion process $(x_t)_{t \geq 0}$ and model the entire trajectories:

With a time reversal, we can define the process just from **scores** rather than densities:

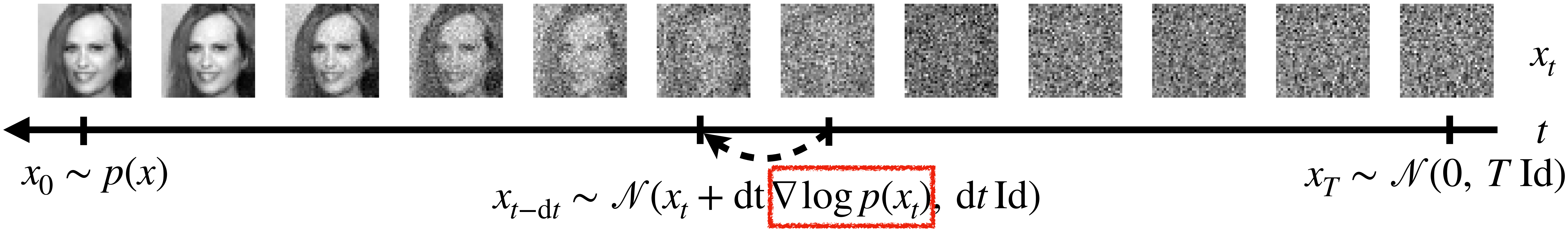


(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

Diffusion models to the rescue

Introduce a diffusion process $(x_t)_{t\geq 0}$ and model the entire trajectories:

With a time reversal, we can define the process just from **scores** rather than densities:



(Anderson, 1982; Sohl-Dickstein et al., 2015; Song et al., 2019, 2020; Ho et al., 2020; Kadkhodaie & Simoncelli, 2020)

From density to scores and back

- Scores do not depend on normalizing constants: easy to learn

$$-\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

$$-\nabla_x \log p_\theta(x) = \nabla_x U_\theta(x)$$

From density to scores and back

- Scores do not depend on normalizing constants: easy to learn

$$-\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

$$-\nabla_x \log p_\theta(x) = \nabla_x U_\theta(x) \longrightarrow$$

Amounts to denoising (regression)

Learn score models $s_\theta(x_t, t)$

From density to scores and back

- Scores do not depend on normalizing constants: easy to learn

$$-\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

$$-\nabla_x \log p_\theta(x) = \nabla_x U_\theta(x) \longrightarrow$$

Amounts to denoising (regression)

Learn score models $s_\theta(x_t, t)$

- **All times are needed:** the score at time 0 **does *not* constrain** the density sufficiently!

From density to scores and back

- Scores do not depend on normalizing constants: easy to learn

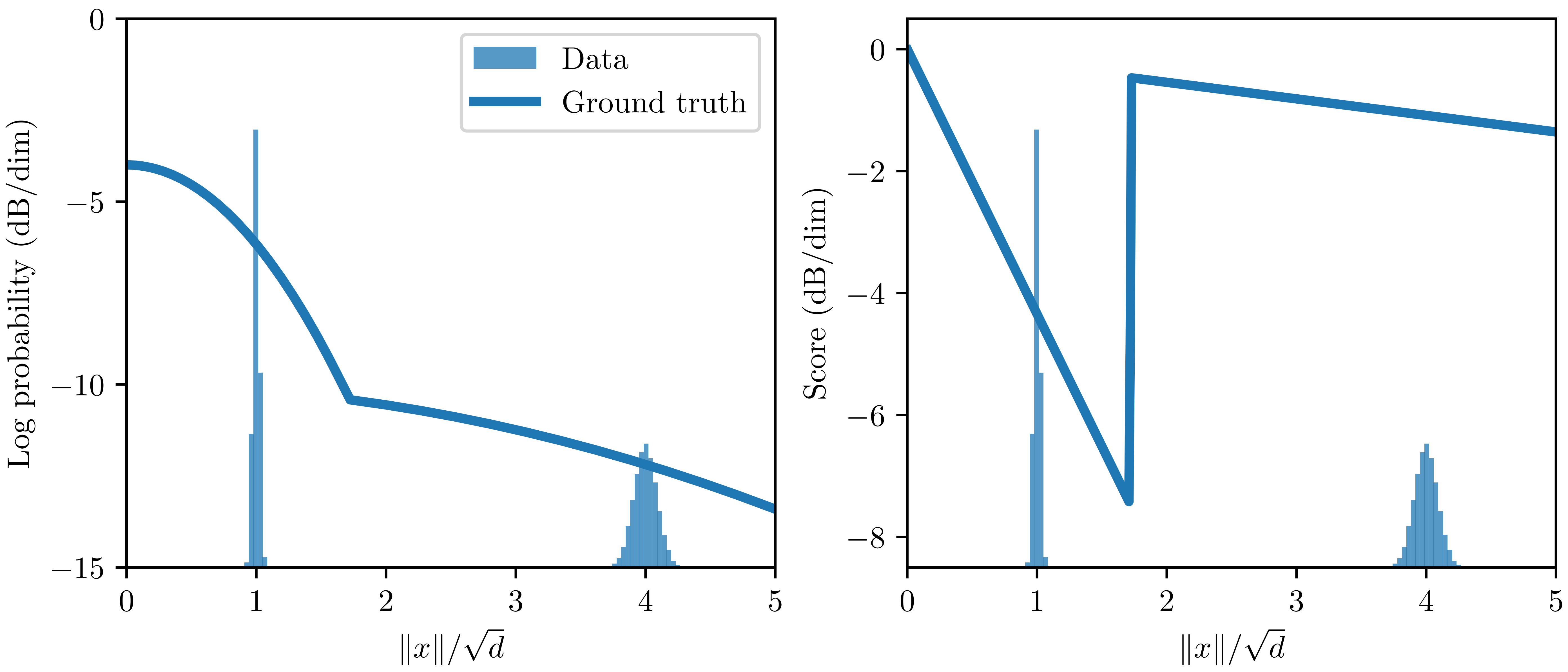
$$-\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

$$-\nabla_x \log p_\theta(x) = \nabla_x U_\theta(x) \longrightarrow$$

Amounts to denoising (regression)

Learn score models $s_\theta(x_t, t)$

- **All times are needed:** the score at time 0 **does *not* constrain** the density sufficiently!



From density to scores and back

- Scores do not depend on normalizing constants: easy to learn

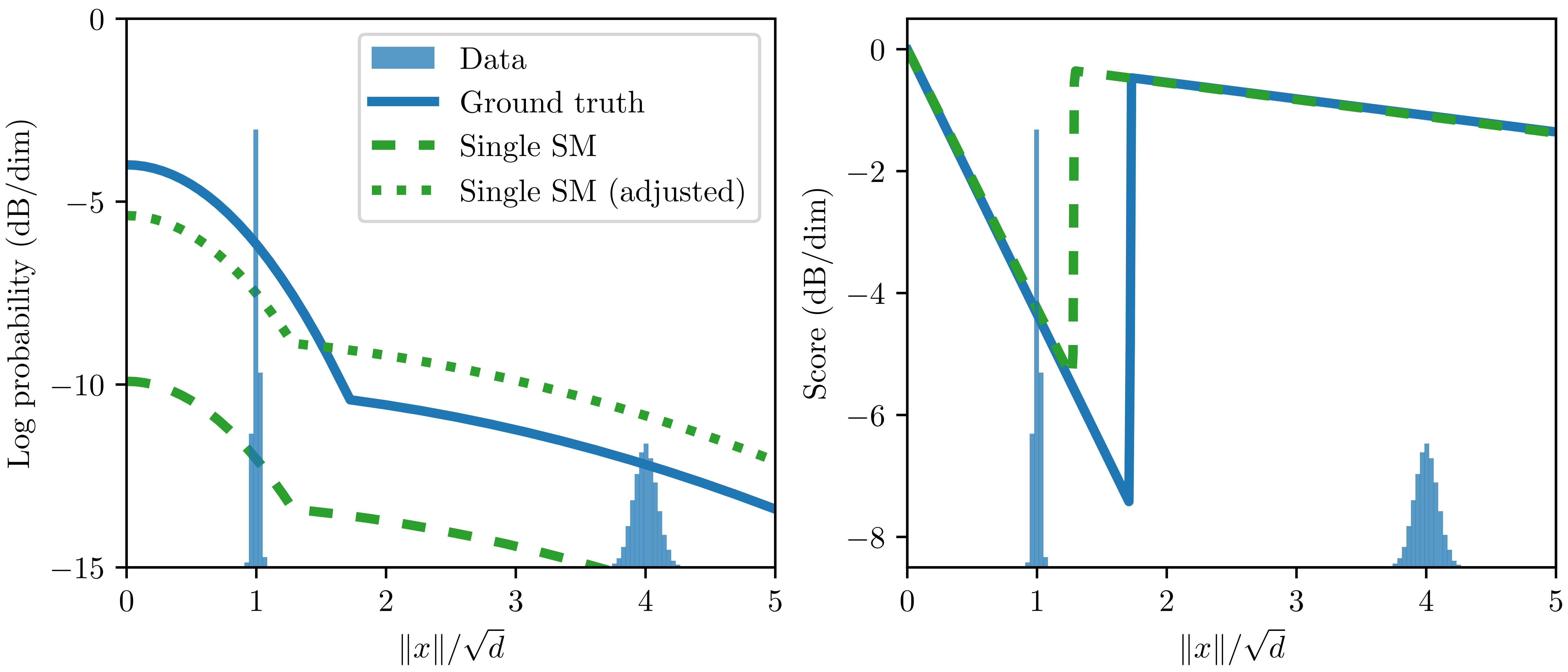
$$-\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

$$-\nabla_x \log p_\theta(x) = \nabla_x U_\theta(x) \longrightarrow$$

Amounts to denoising (regression)

Learn score models $s_\theta(x_t, t)$

- **All times are needed:** the score at time 0 **does *not* constrain** the density sufficiently!



From density to scores and back

- Scores do not depend on normalizing constants: easy to learn

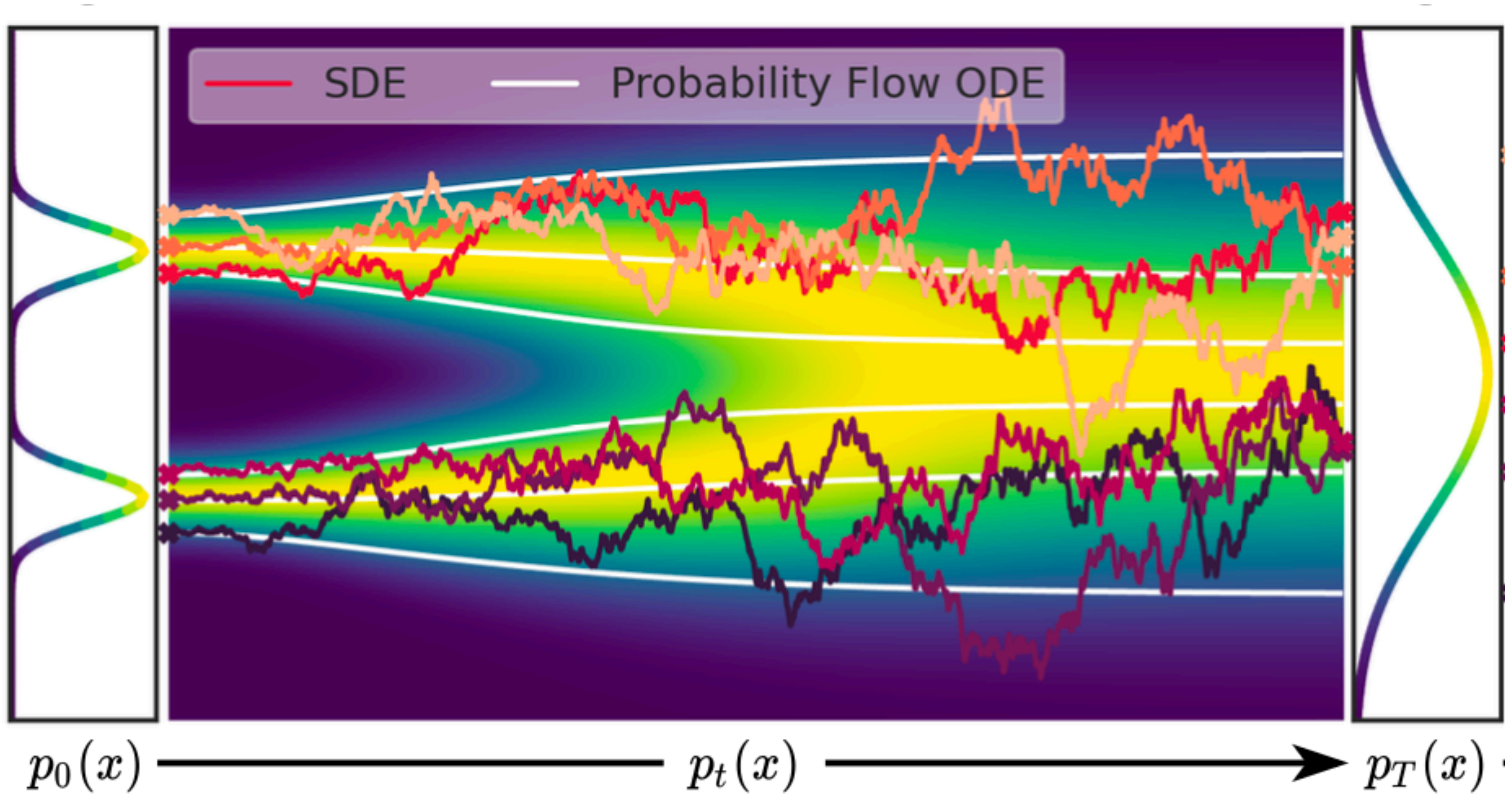
$$-\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

$$-\nabla_x \log p_\theta(x) = \nabla_x U_\theta(x) \longrightarrow$$

Amounts to denoising (regression)

Learn score models $s_\theta(x_t, t)$

- **All times are needed:** the score at time 0 **does *not* constrain** the density sufficiently!



From density to scores and back

- Scores do not depend on normalizing constants: easy to learn

$$-\log p_\theta(x) = U_\theta(x) + \log Z_\theta$$

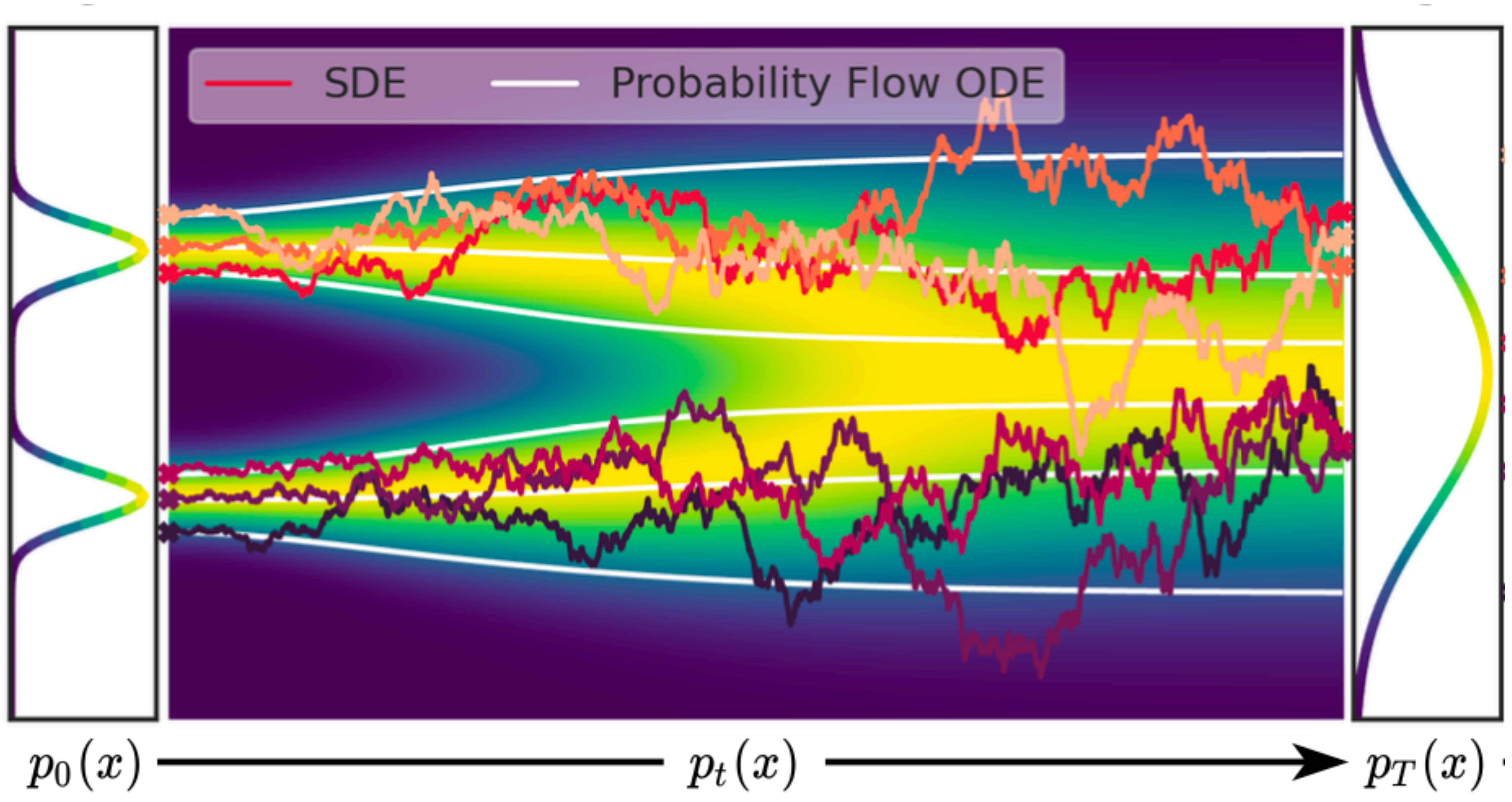
$$-\nabla_x \log p_\theta(x) = \nabla_x U_\theta(x) \longrightarrow$$

Amounts to denoising (regression)

Learn score models $s_\theta(x_t, t)$

- **All times are needed:** the score at time 0 **does *not* constrain** the density sufficiently!

$p_\theta(x)$ can be recovered from the score family but expensive (integration along time, …). Is there a more direct approach?

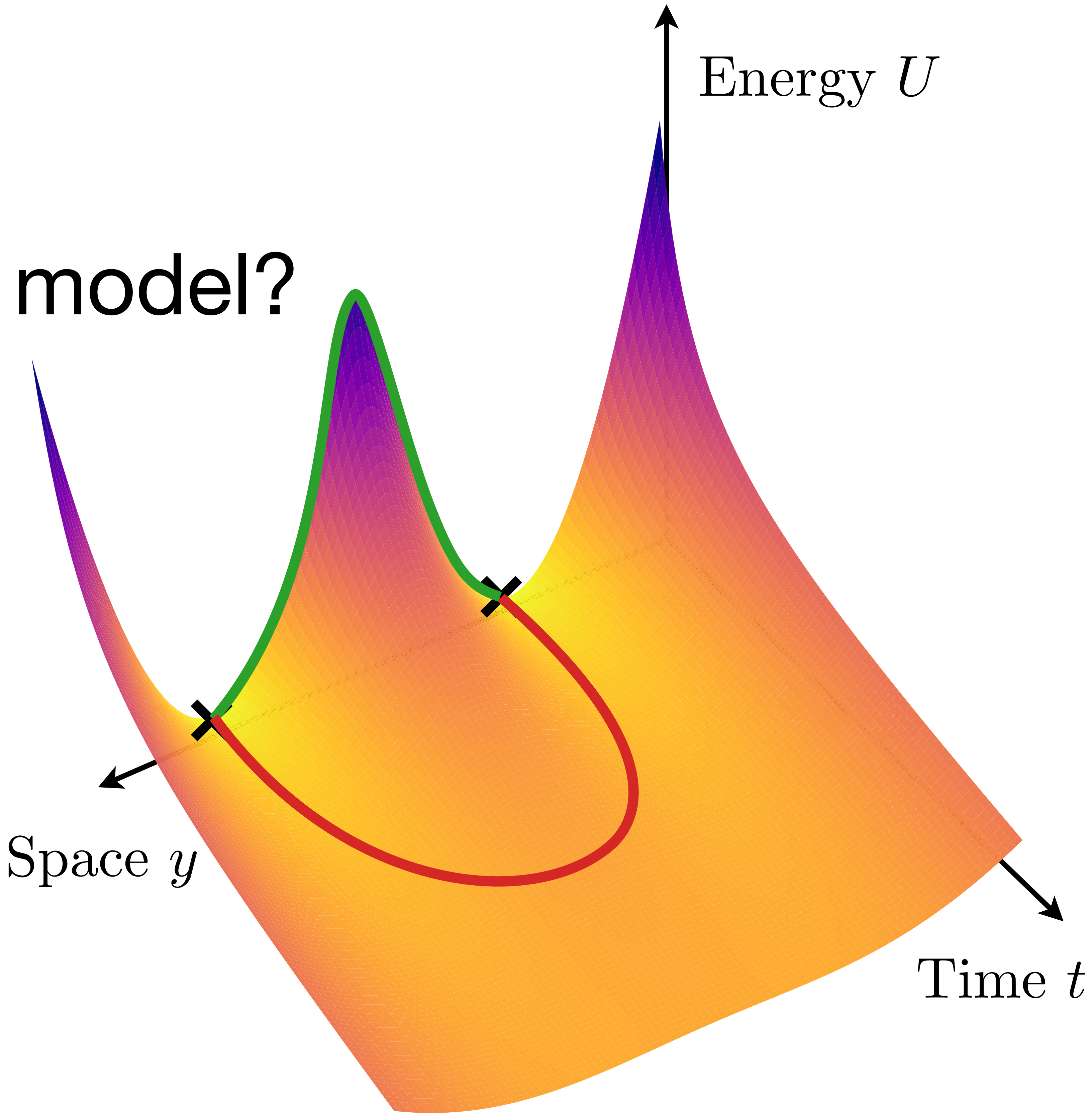


Dual score matching

- How to use ideas from diffusion to get an explicit energy model?

Dual score matching

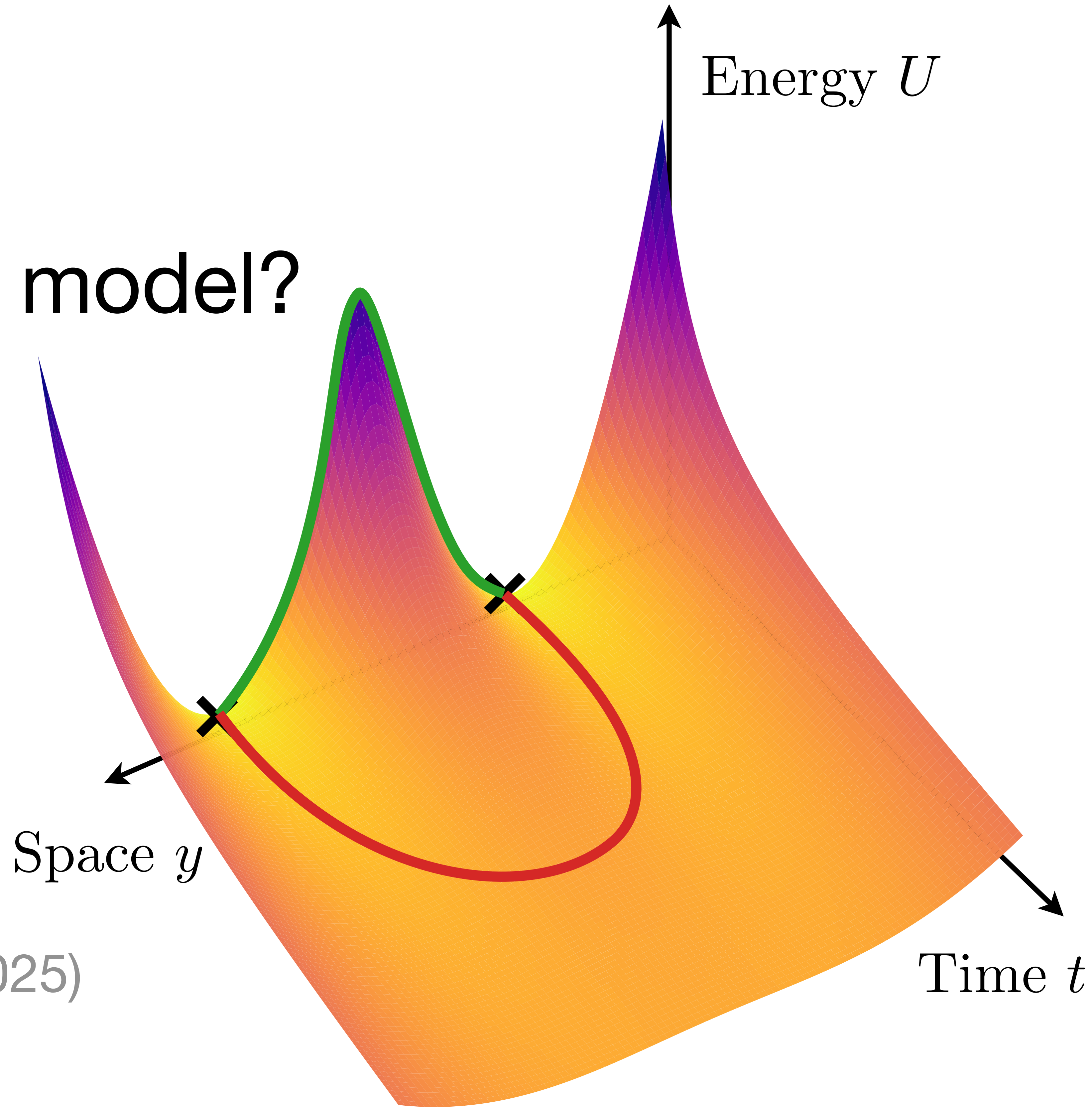
- How to use ideas from diffusion to get an explicit energy model?
- x can be multimodal, but (y, t) is roughly unimodal! where $y = x + \sqrt{t}z$



Dual score matching

- How to use ideas from diffusion to get an explicit energy model?
- x can be multimodal, but (y, t) is roughly unimodal! where $y = x + \sqrt{t}z$
- Let's parameterize $U(y, t)$ with a network and do joint score matching on (y, t)!
- That is, score matching on $\nabla_y U(y, t)$ and $\partial_t U(y, t)$

(Choi et al., 2022; Yadin et al., 2024; Yu et al.; 2025)

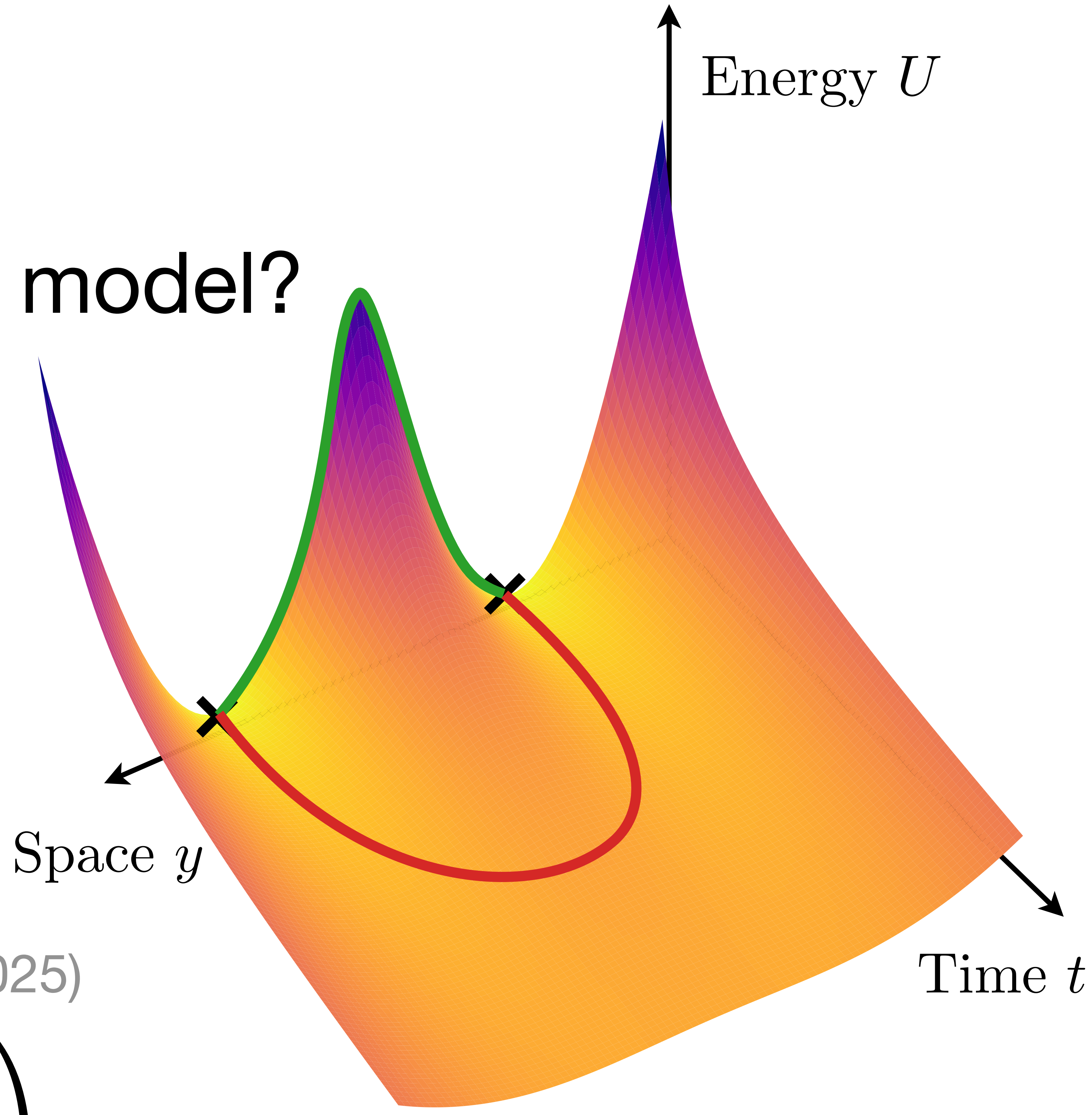


Dual score matching

- How to use ideas from diffusion to get an explicit energy model?
- x can be multimodal, but (y, t) is roughly unimodal!
  where $y = x + \sqrt{t}z$
- Let's parameterize $U(y, t)$ with a network and do joint score matching on (y, t)!
- That is, score matching on $\nabla_y U(y, t)$ and $\partial_t U(y, t)$

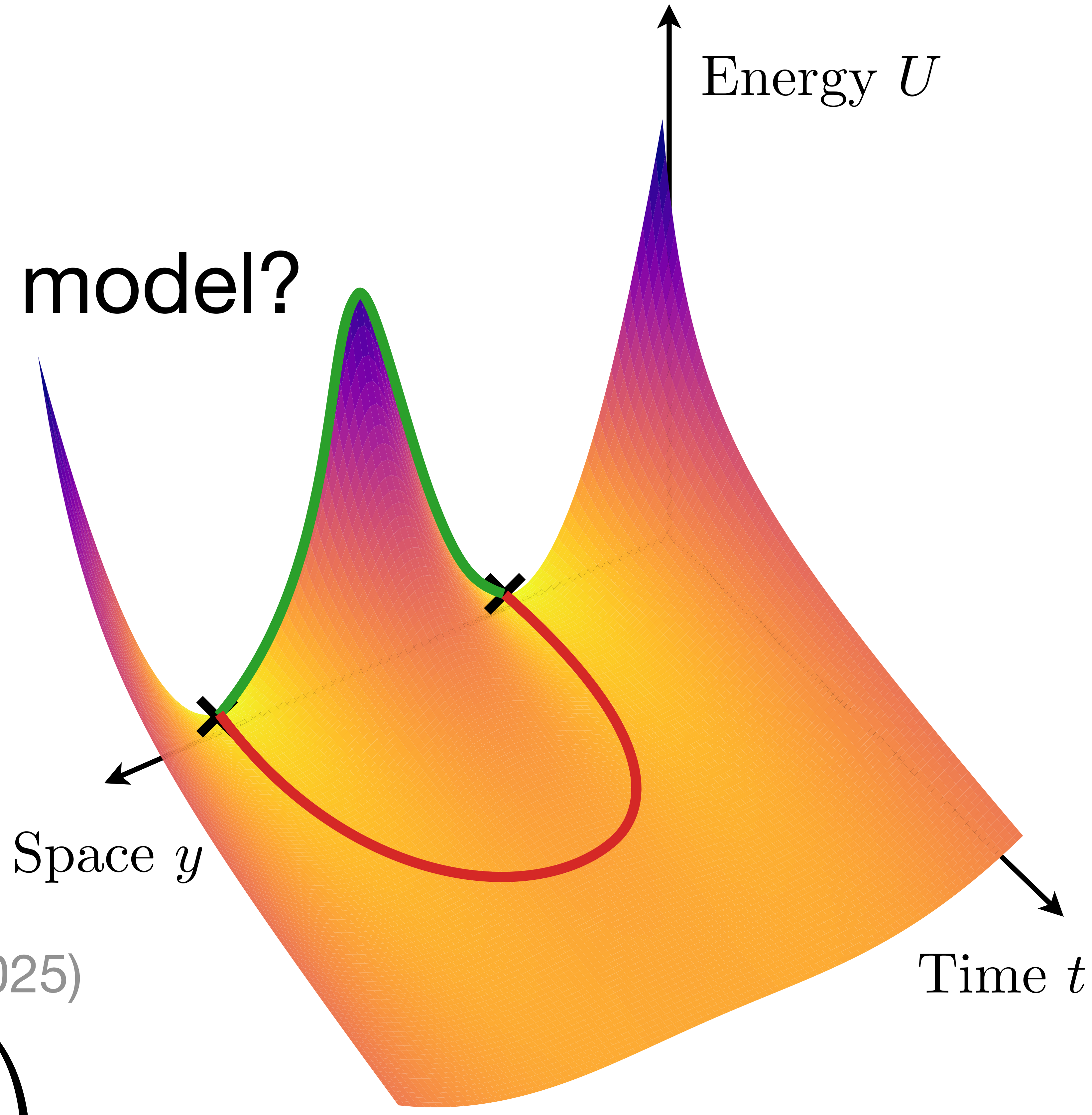
(Choi et al., 2022; Yadin et al., 2024; Yu et al.; 2025)

$$U(y, t) = -\log\left(\int p(x)\mathrm{e}^{-\frac{1}{2t}\|y-x\|^2-\frac{d}{2}\log(2\pi t)}\mathrm{d}x\right)$$



Dual score matching

- How to use ideas from diffusion to get an explicit energy model?
- x can be multimodal, but (y,t) is roughly unimodal! where $y = x + \sqrt{t}z$
- Let's parameterize $U(y,t)$ with a network and do joint score matching on (y,t)!
- That is, score matching on $\nabla_y U(y,t)$ and $\partial_t U(y,t)$



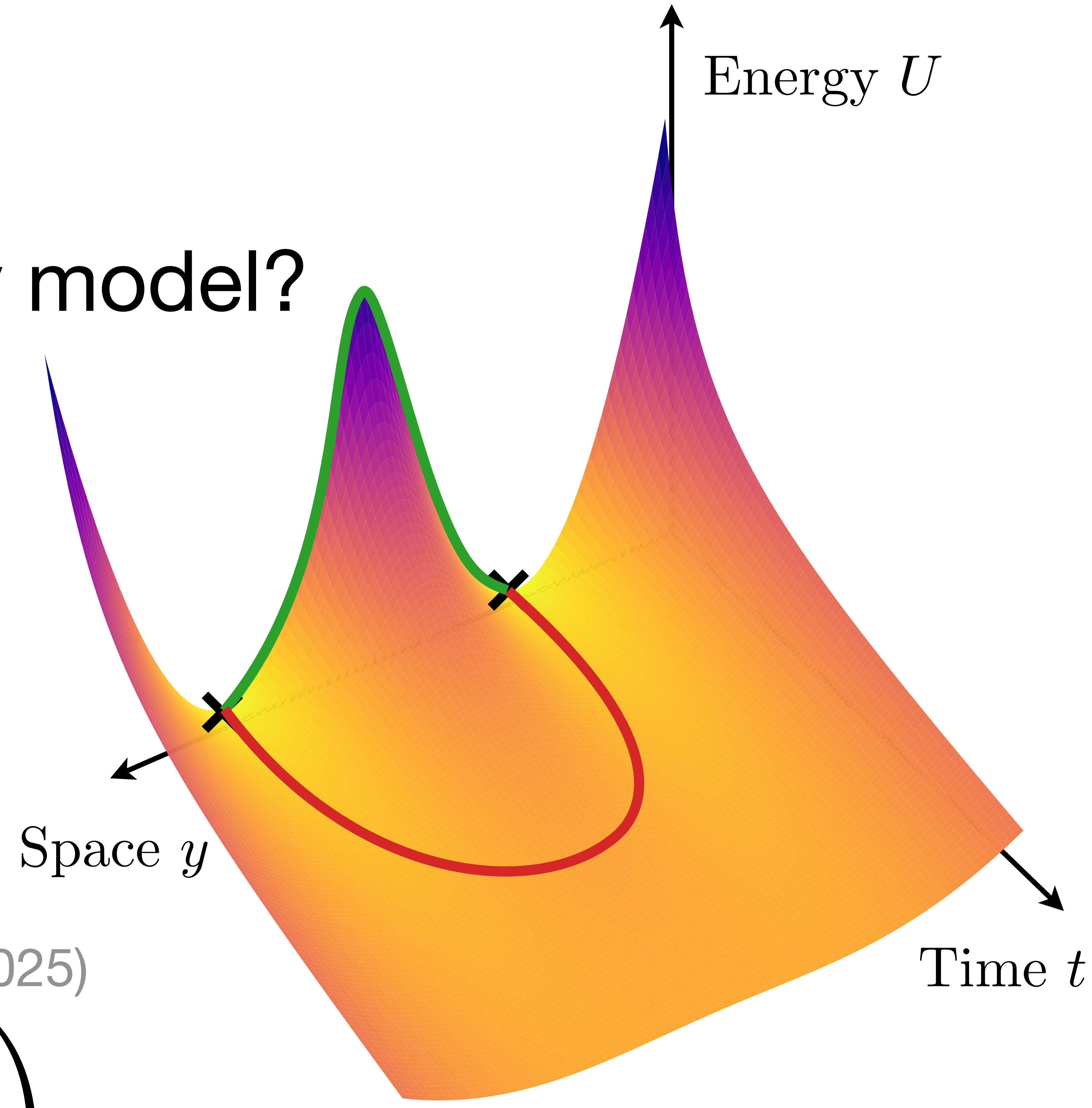
(Choi et al., 2022; Yadin et al., 2024; Yu et al.; 2025)

$$U(y,t) = -\log\left(\int p(x) \mathrm{e}^{-\frac{1}{2t}\|y-x\|^2 - \frac{d}{2}\log(2\pi t)} \mathrm{d}x\right)$$

$$\nabla_y U(y,t) = \mathbb{E}_x\left[\frac{y-x}{t} \,|\, y\right]$$

Dual score matching

- How to use ideas from diffusion to get an explicit energy model?
- x can be multimodal, but (y, t) is roughly unimodal! where $y = x + \sqrt{t}z$
- Let's parameterize $U(y, t)$ with a network and do joint score matching on (y, t)!
- That is, score matching on $\nabla_y U(y, t)$ and $\partial_t U(y, t)$



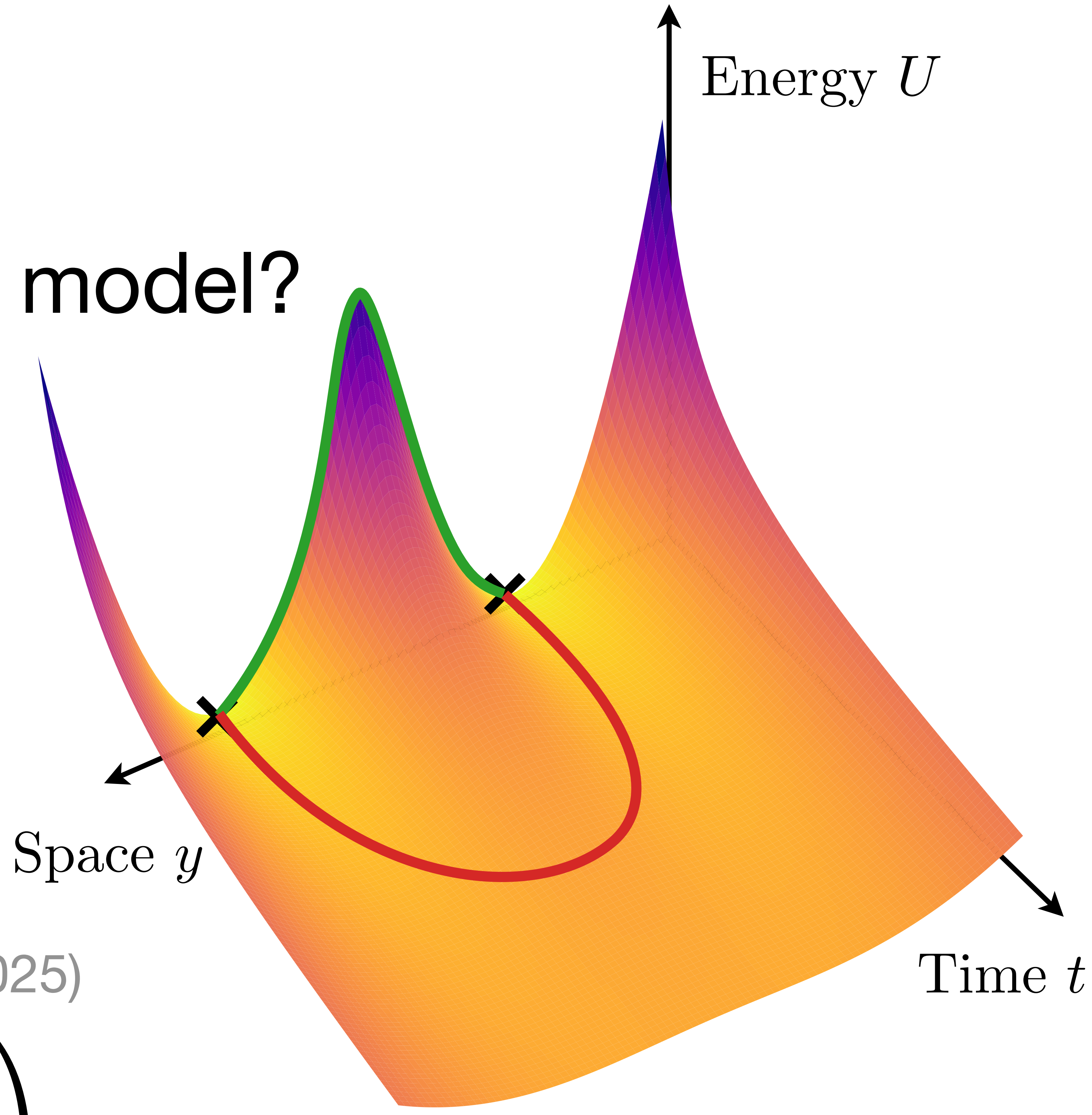
(Choi et al., 2022; Yadin et al., 2024; Yu et al.; 2025)

$$U(y, t) = -\log\left(\int p(x) \mathrm{e}^{-\frac{1}{2t}\|y-x\|^2 - \frac{d}{2}\log(2\pi t)} \mathrm{d}x\right)$$

$$\nabla_y U(y, t) = \mathbb{E}_x\left[\frac{y - x}{t} \,\middle|\, y\right] \longrightarrow \ell_{\mathrm{DSM}}(\theta, t) = \mathbb{E}_{x,y}\left[\|\nabla_y U_\theta(y, t) - \frac{y - x}{t}\|^2\right]$$

Dual score matching

- How to use ideas from diffusion to get an explicit energy model?
- x can be multimodal, but (y, t) is roughly unimodal! where $y = x + \sqrt{t}z$
- Let's parameterize $U(y, t)$ with a network and do joint score matching on (y, t)!
- That is, score matching on $\nabla_y U(y, t)$ and $\partial_t U(y, t)$



(Choi et al., 2022; Yadin et al., 2024; Yu et al.; 2025)

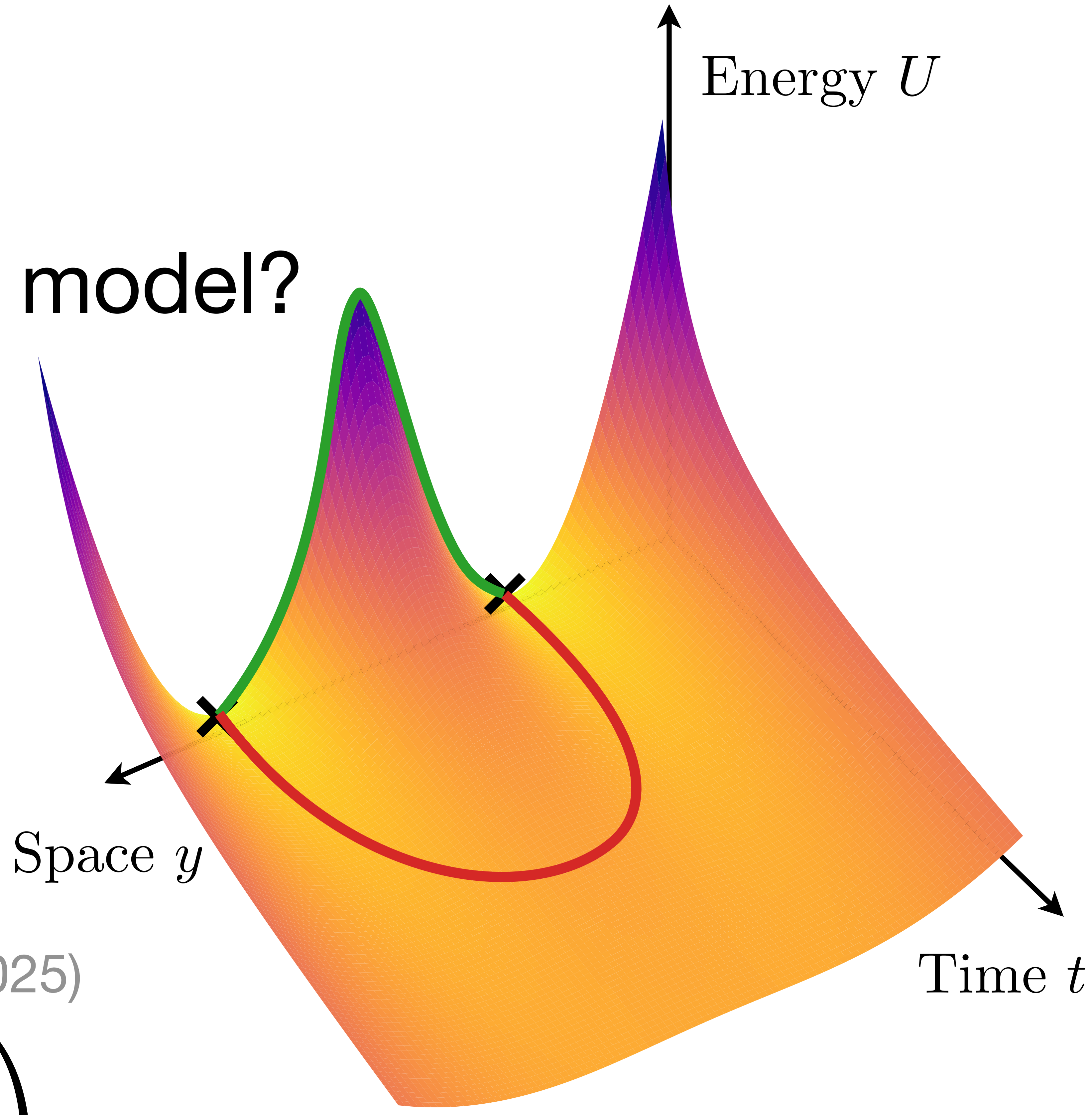
$$U(y, t) = -\log\left(\int p(x) \mathrm{e}^{-\frac{1}{2t}\|y-x\|^2 - \frac{d}{2}\log(2\pi t)} \mathrm{d}x\right)$$

$$\nabla_y U(y, t) = \mathbb{E}_x\left[\frac{y - x}{t} \,|\, y\right] \longrightarrow \ell_{\mathrm{DSM}}(\theta, t) = \mathbb{E}_{x,y}\left[\|\nabla_y U_\theta(y, t) - \frac{y - x}{t}\|^2\right]$$

$$\partial_t U(y, t) = \mathbb{E}_x\left[\frac{d}{2t} - \frac{\|y - x\|^2}{2t^2} \,|\, y\right]$$

Dual score matching

- How to use ideas from diffusion to get an explicit energy model?
- x can be multimodal, but (y, t) is roughly unimodal! where $y = x + \sqrt{t}z$
- Let's parameterize $U(y, t)$ with a network and do joint score matching on (y, t)!
- That is, score matching on $\nabla_y U(y, t)$ and $\partial_t U(y, t)$



(Choi et al., 2022; Yadin et al., 2024; Yu et al.; 2025)

$$U(y,t) = -\log\left(\int p(x)\mathrm{e}^{-\frac{1}{2t}\|y-x\|^2 - \frac{d}{2}\log(2\pi t)}\mathrm{d}x\right)$$

$$\nabla_y U(y,t) = \mathbb{E}_x\left[\frac{y-x}{t} \,|\, y\right] \longrightarrow \ell_{\mathrm{DSM}}(\theta,t) = \mathbb{E}_{x,y}\left[\|\nabla_y U_\theta(y,t) - \frac{y-x}{t}\|^2\right]$$

$$\partial_t U(y,t) = \mathbb{E}_x\left[\frac{d}{2t} - \frac{\|y-x\|^2}{2t^2} \,|\, y\right] \longrightarrow \ell_{\mathrm{TSM}}(\theta,t) = \mathbb{E}_{x,y}\left[\left(\partial_t U_\theta(y,t) - \frac{d}{2t} + \frac{\|y-x\|^2}{2t^2}\right)^2\right]$$

Dual score matching

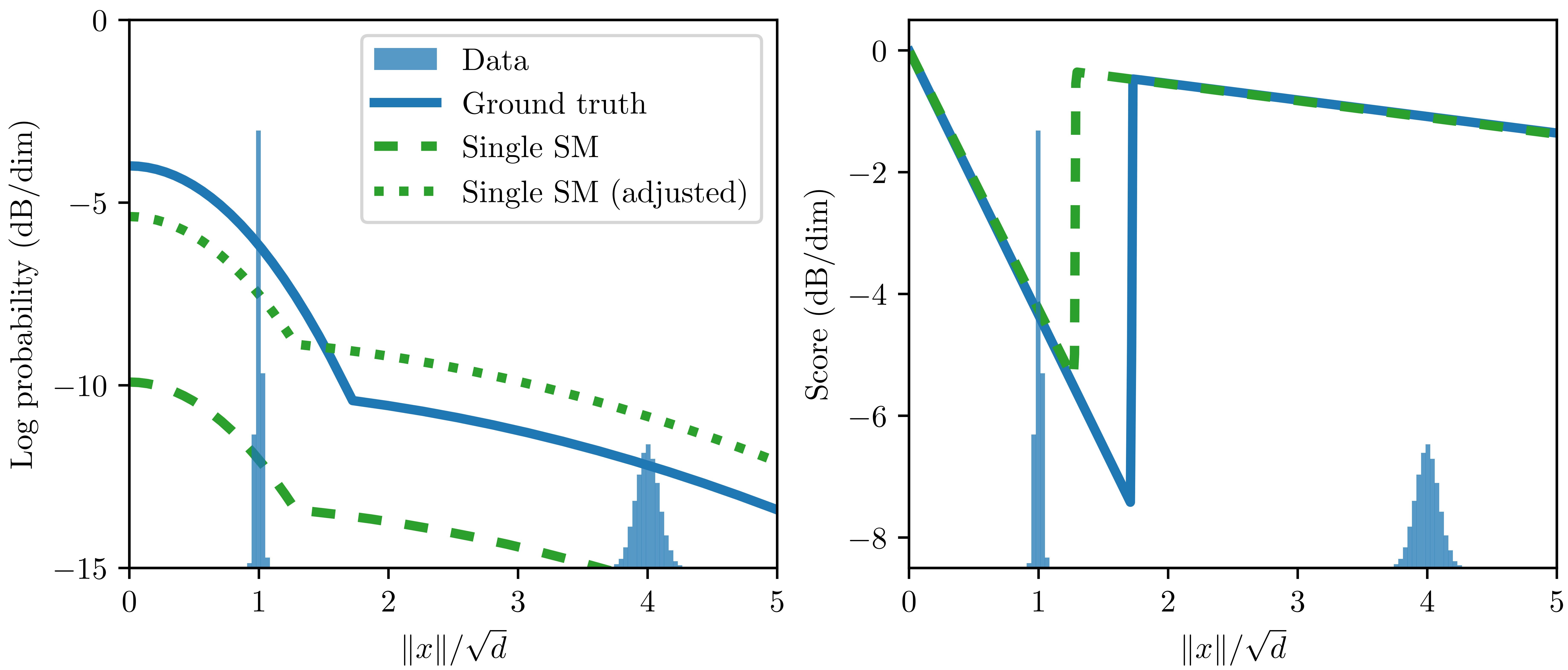
- Optimize the sum, integrated over t: $\ell(\theta) = \mathbb{E}_t \left[\frac{t}{d} \ell_{\mathrm{DSM}}(\theta, t) + \left(\frac{t}{d} \right)^2 \ell_{\mathrm{TSM}}(\theta, t) \right]$

Dual score matching

- Optimize the sum, integrated over t: $\ell(\theta) = \mathbb{E}_t \left[\frac{t}{d} \ell_{\mathrm{DSM}}(\theta, t) + \left(\frac{t}{d} \right)^2 \ell_{\mathrm{TSM}}(\theta, t) \right]$
- **Post-training normalization: mass conservation** + nearly Gaussian at large t

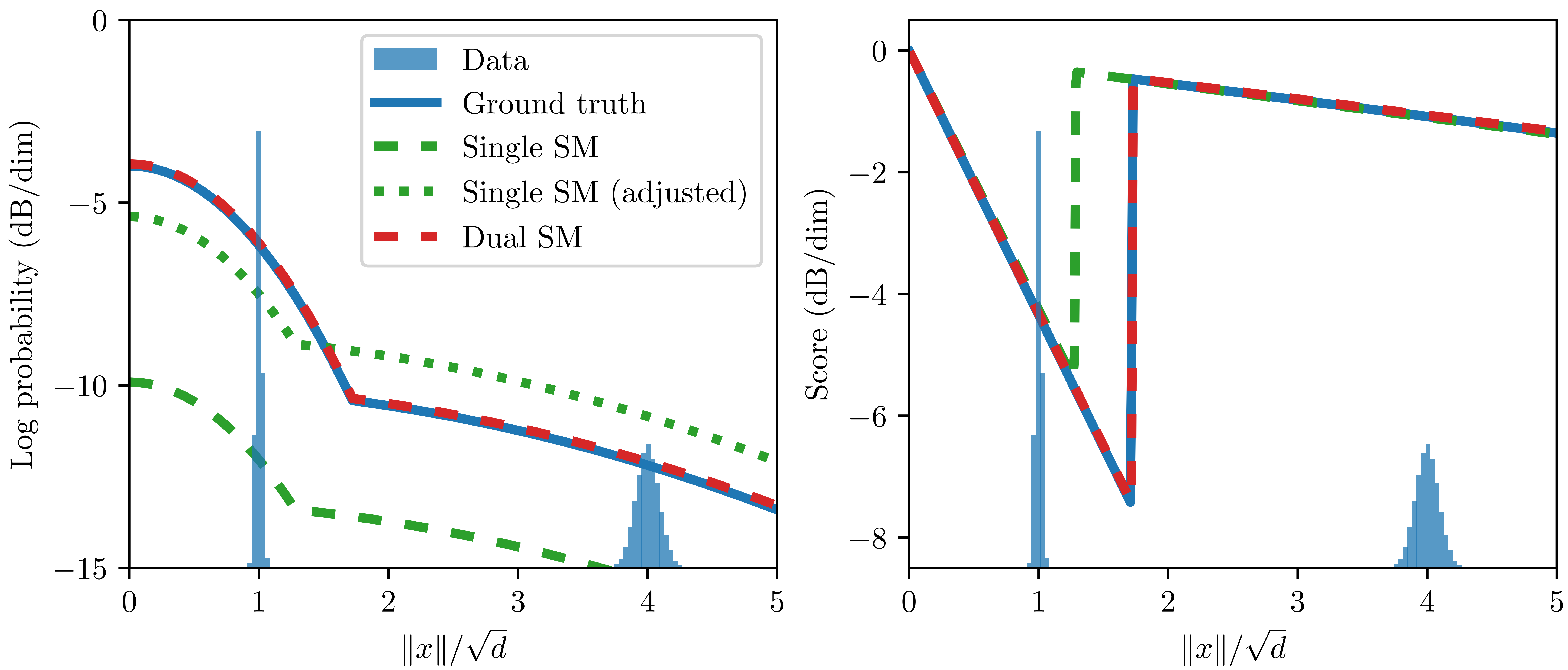
Dual score matching

- Optimize the sum, integrated over t: $\ell(\theta) = \mathbb{E}_t\left[\frac{t}{d}\ell_{\mathrm{DSM}}(\theta,t) + \left(\frac{t}{d}\right)^2 \ell_{\mathrm{TSM}}(\theta,t)\right]$
- **Post-training normalization: mass conservation** + nearly Gaussian at large t



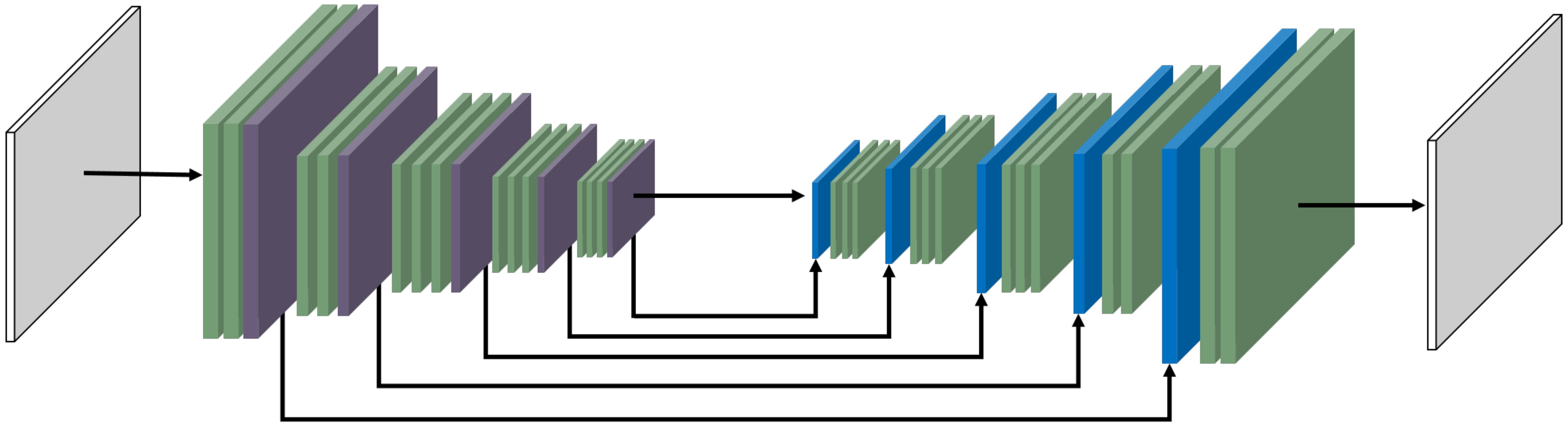
Dual score matching

- Optimize the sum, integrated over t: $\ell(\theta) = \mathbb{E}_t \left[\frac{t}{d} \ell_{\mathrm{DSM}}(\theta, t) + \left(\frac{t}{d}\right)^2 \ell_{\mathrm{TSM}}(\theta, t) \right]$
- **Post-training normalization: mass conservation** + nearly Gaussian at large t



What architecture for an energy model?

We know a good architecture for the score (UNet).

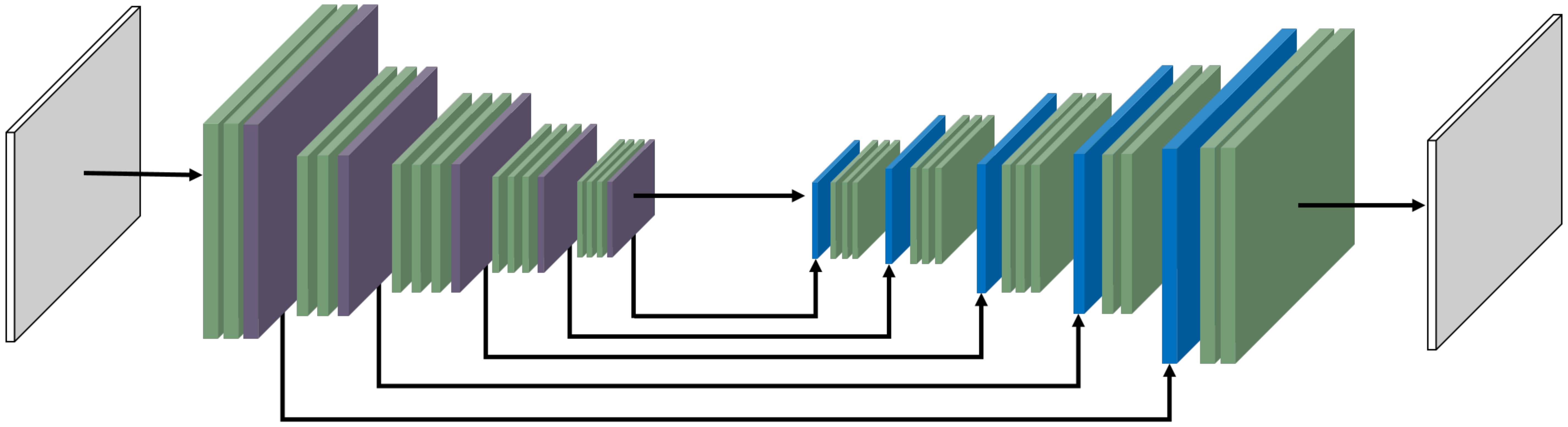


How to preserve its inductive biases when learning the energy?

(Romano et al., 2017; Mohan*, Kadkhodaie*, et al., 2020; Salimans and Ho, 2021; Hurault et al., 2021; Du et al., 2023; Yadin et al., 2024; Thornton et al., 2025)

What architecture for an energy model?

We know a good architecture for the score (UNet).



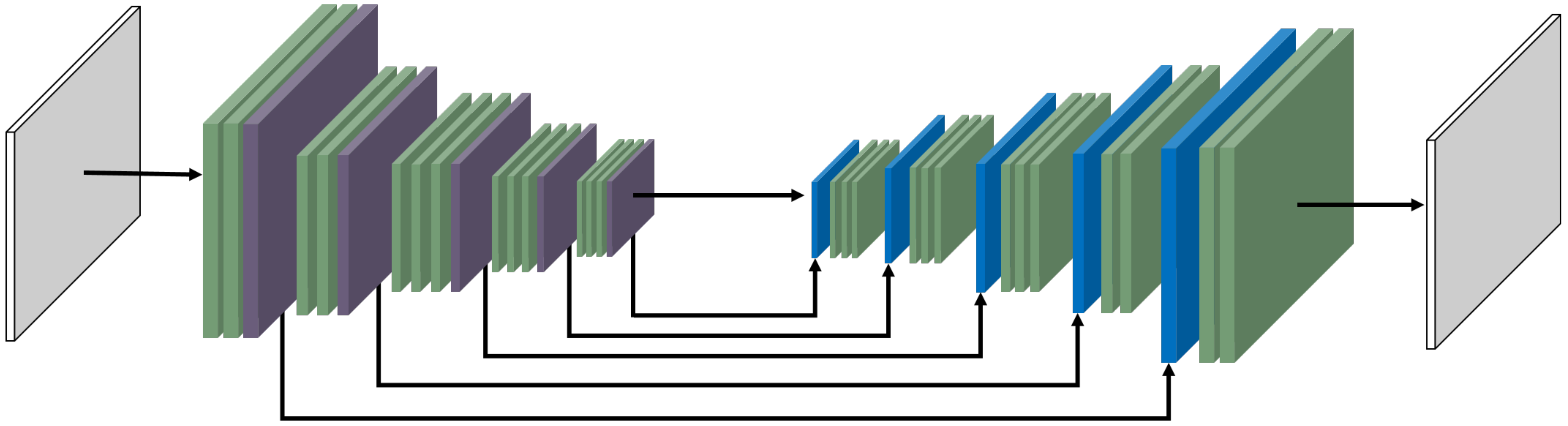
How to preserve its inductive biases when learning the energy?

That is, we would like $\nabla_y U_\theta(y, t) = s_\theta(y, t)$

(Romano et al., 2017; Mohan*, Kadkhodaie*, et al., 2020; Salimans and Ho, 2021; Hurault et al., 2021; Du et al., 2023; Yadin et al., 2024; Thornton et al., 2025)

What architecture for an energy model?

We know a good architecture for the score (UNet).



How to preserve its inductive biases when learning the energy?

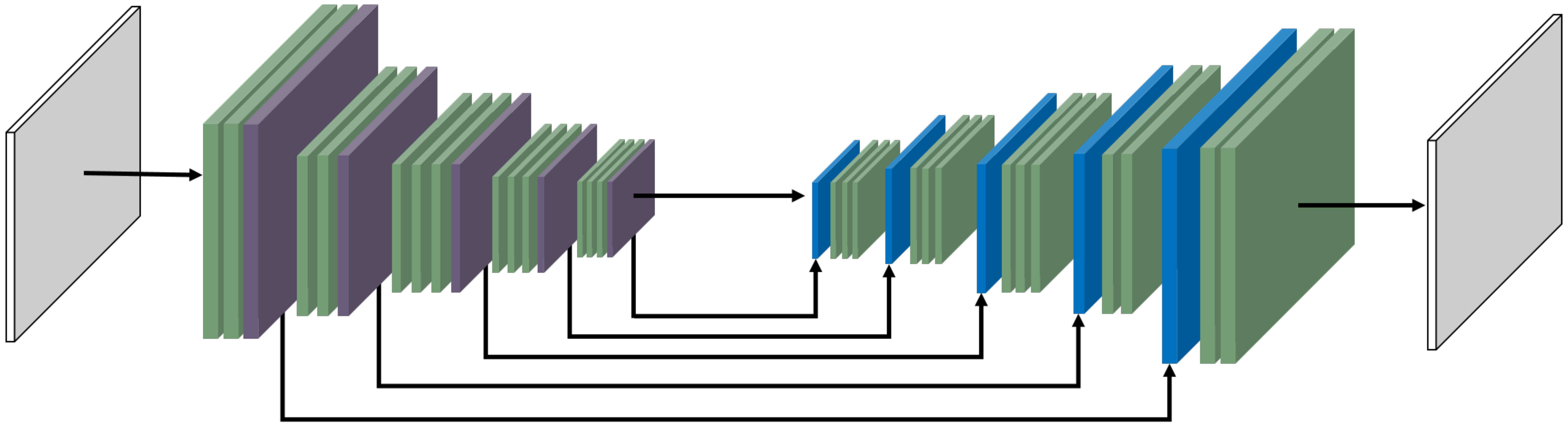
That is, we would like $\nabla_y U_\theta(y, t) = s_\theta(y, t)$

Define $$U_\theta(y, t) = \frac{1}{2}\langle y, s_\theta(y, t)\rangle$$

(Romano et al., 2017; Mohan*, Kadkhodaie*, et al., 2020; Salimans and Ho, 2021; Hurault et al., 2021; Du et al., 2023; Yadin et al., 2024; Thornton et al., 2025)

What architecture for an energy model?

We know a good architecture for the score (UNet).



How to preserve its inductive biases when learning the energy?

That is, we would like $\nabla_y U_\theta(y,t) = s_\theta(y,t)$

Define $$U_\theta(y,t) = \frac{1}{2}\langle y, s_\theta(y,t)\rangle$$

OK if $s_\theta(y,t)$ is **conservative** and **homogeneous**

(Romano et al., 2017; Mohan*, Kadkhodaie*, et al., 2020; Salimans and Ho, 2021; Hurault et al., 2021; Du et al., 2023; Yadin et al., 2024; Thornton et al., 2025)

Let’s train a model on ImageNet!

Evaluating the model

Evaluating the model

- Compare to score model: (slightly) better MSE, same samples

Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

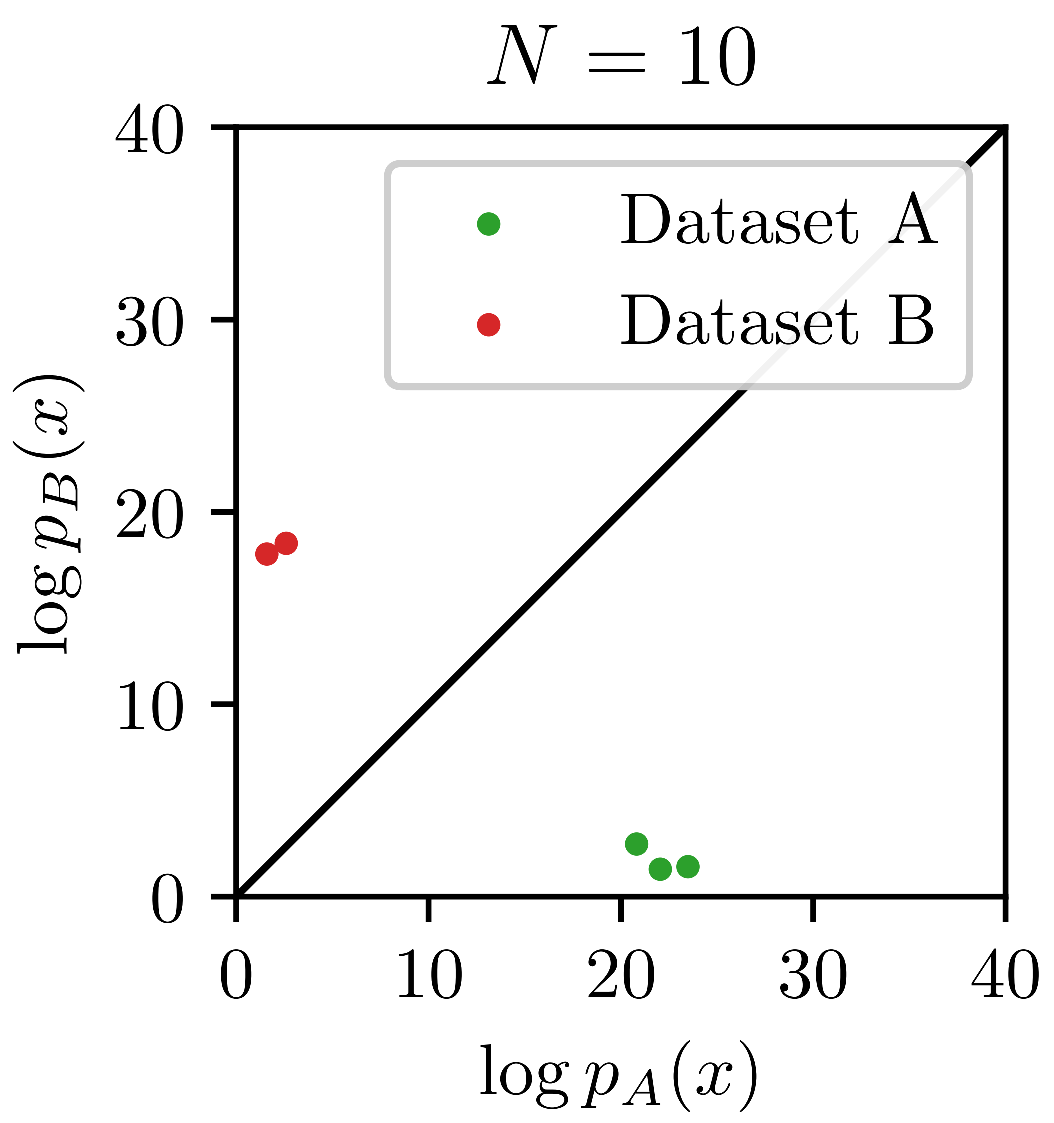
- Train two models on two disjoint sets of N samples
  - Do we learn the same probability model?

Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

- Train two models on two disjoint sets of N samples
  - Do we learn the same probability model?

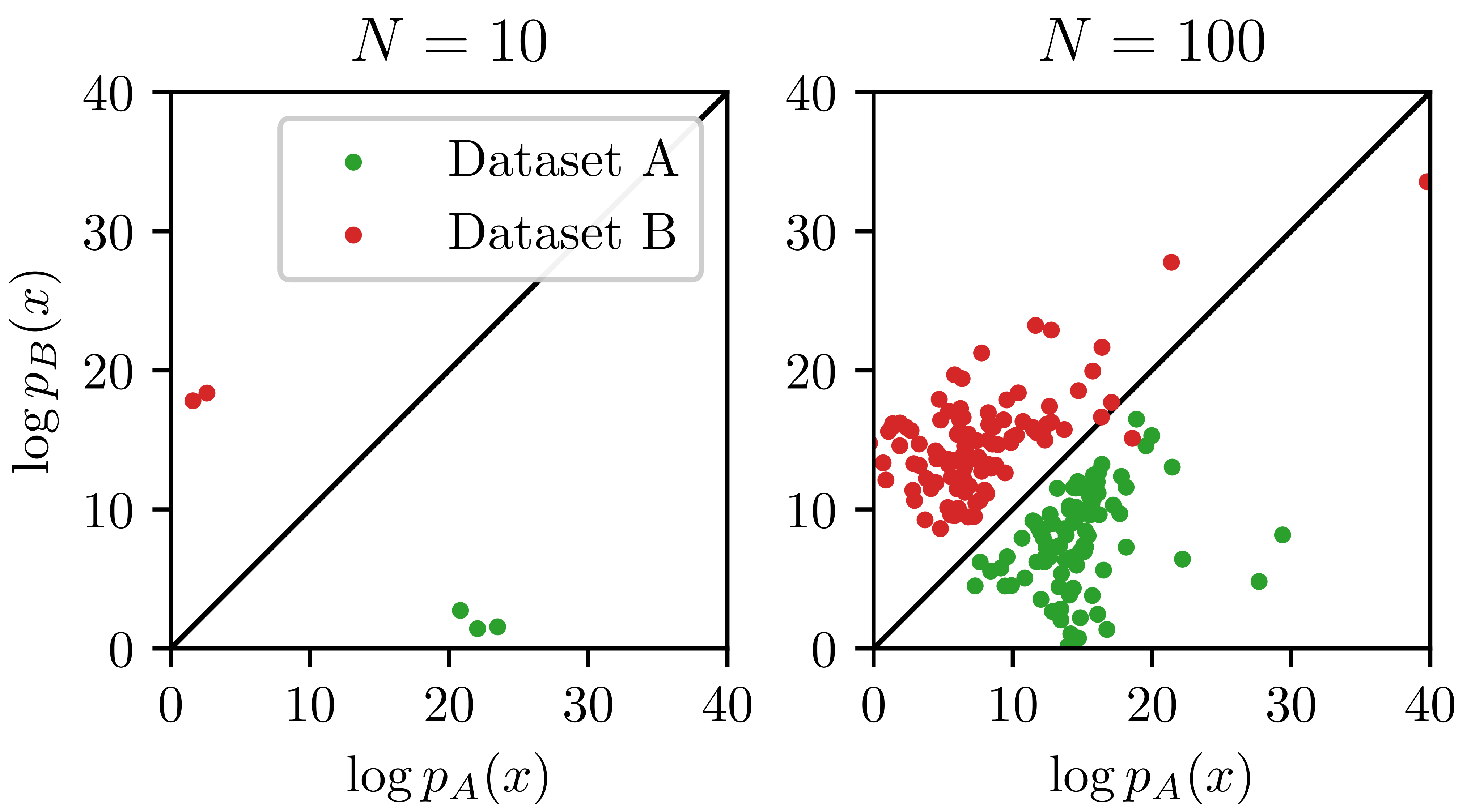


Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

- Train two models on two disjoint sets of N samples
  - Do we learn the same probability model?

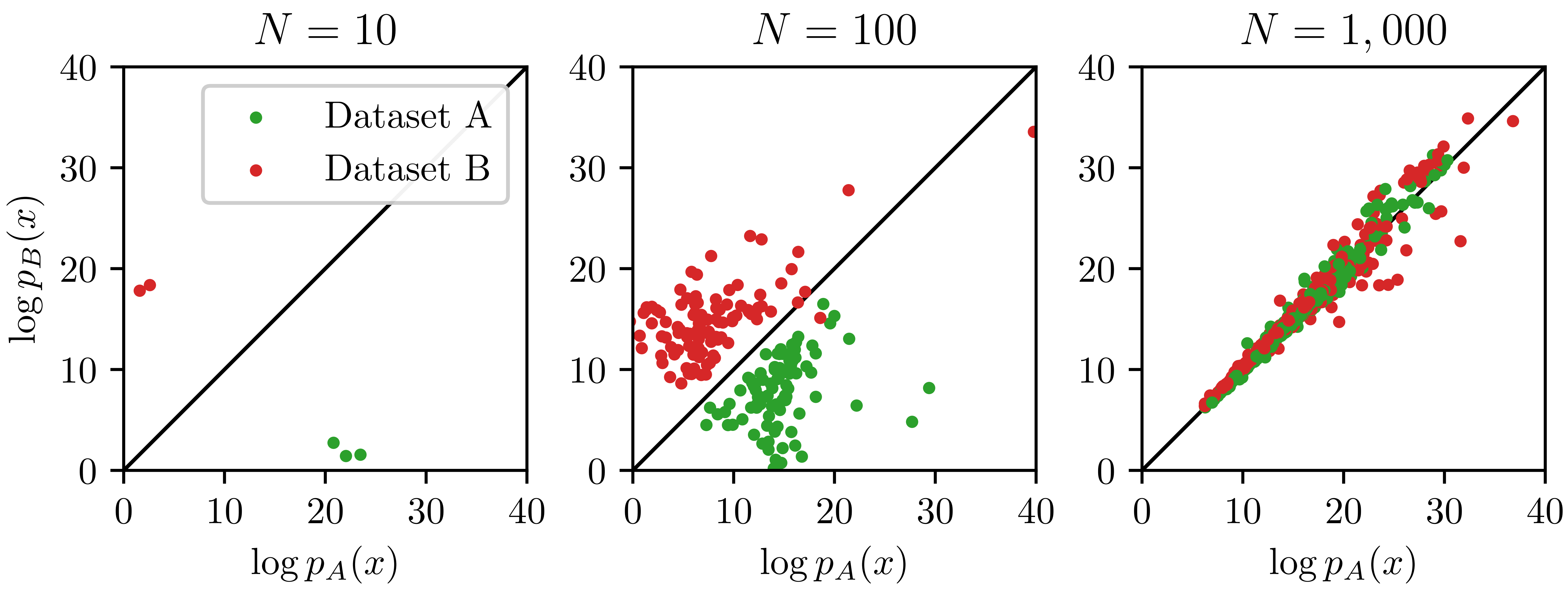


Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

- Train two models on two disjoint sets of N samples
  - Do we learn the same probability model?

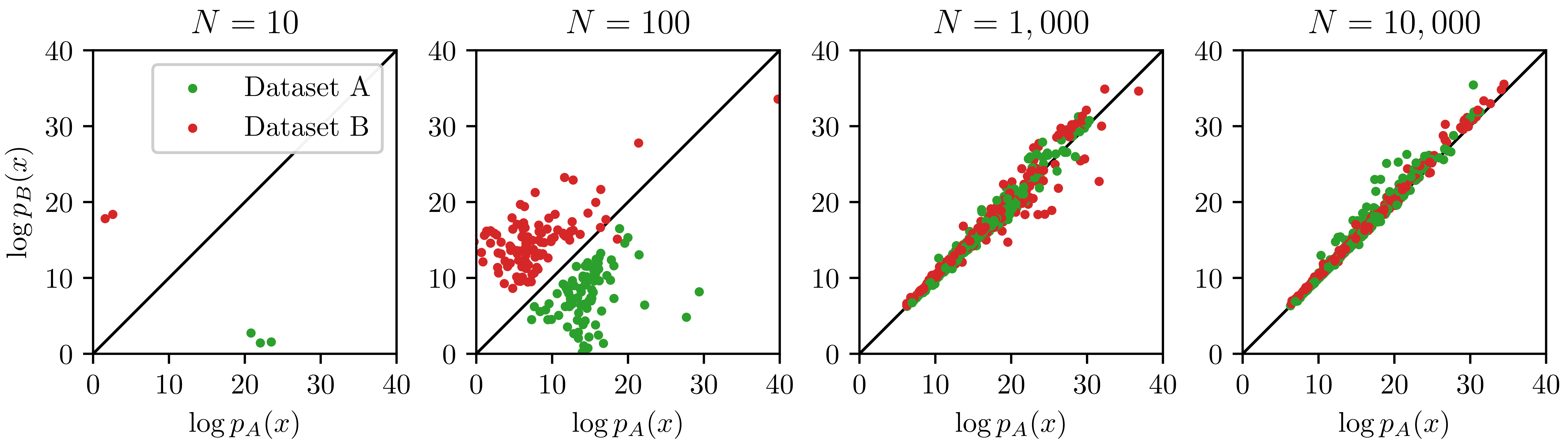


Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

- Train two models on two disjoint sets of N samples
  - Do we learn the same probability model?

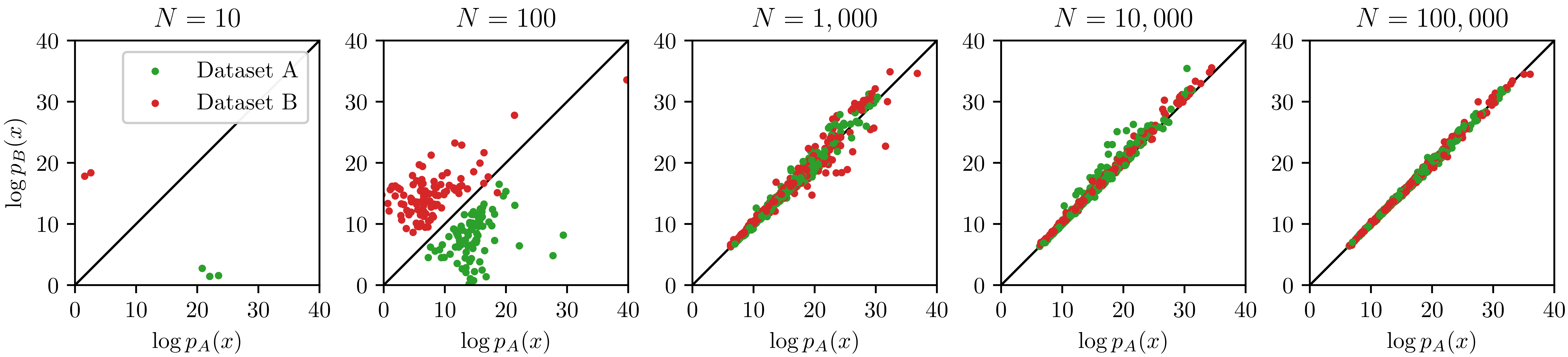


Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

- Train two models on two disjoint sets of N samples
  - Do we learn the same probability model?



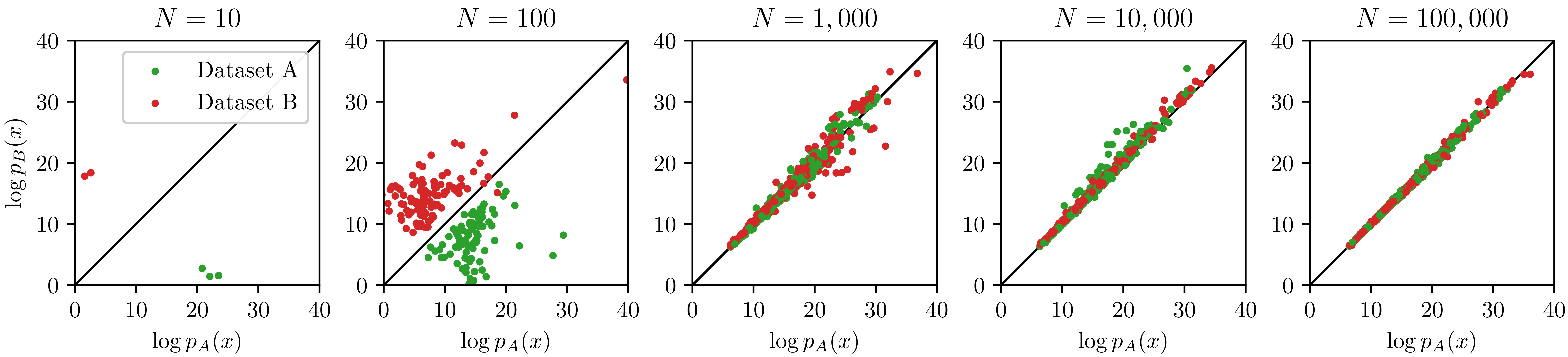
Evaluating the model

- Compare to score model: (slightly) better MSE, same samples
- Cross entropy (negative log likelihood $\mathbb{E}[-\log p_\theta(x)]$) on test set
  - Many caveats: differences between models not informative

(Kong et al., 2023; Karczewski et al., 2025; Bhattacharya et al., 2025)

- Train two models on two disjoint sets of N samples
  - Do we learn the same probability model?

Strong generalization!



The log probability of ImageNet images

Differential entropy: -11.4 dB/dim (roughly volume of $[0, 0.1]^d$)

Quantize: out of $256^d = 10^{9,860}$ possible images, there are $10^{5,180}$ natural images

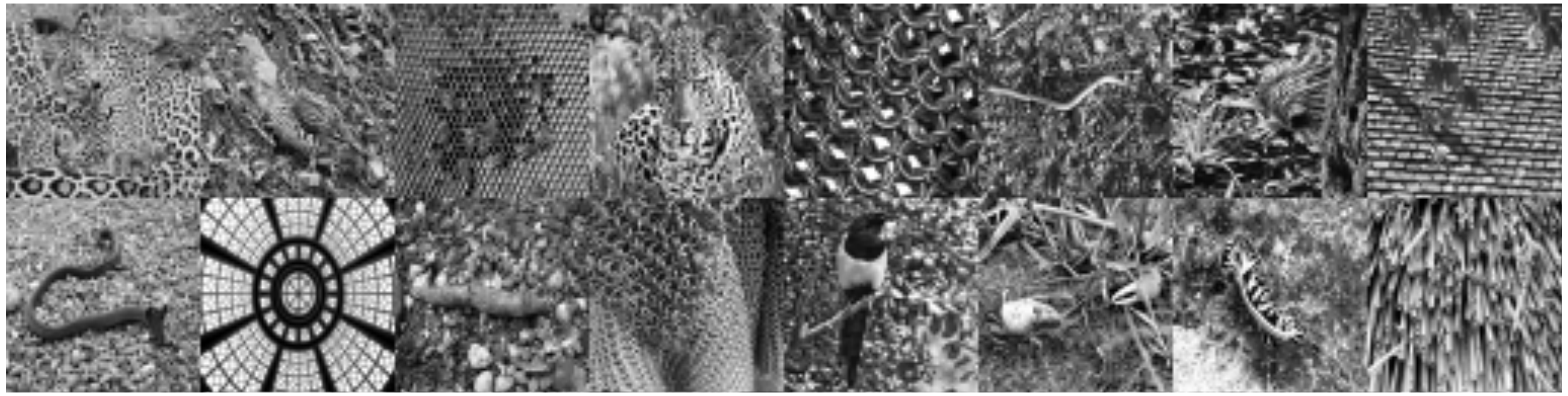
The log probability of ImageNet images

Differential entropy: -11.4 dB/dim (roughly volume of $[0, 0.1]^d$)

Quantize: out of $256^d = 10^{9,860}$ possible images, there are $10^{5,180}$ natural images

Highest probability images:

Lowest probability images:



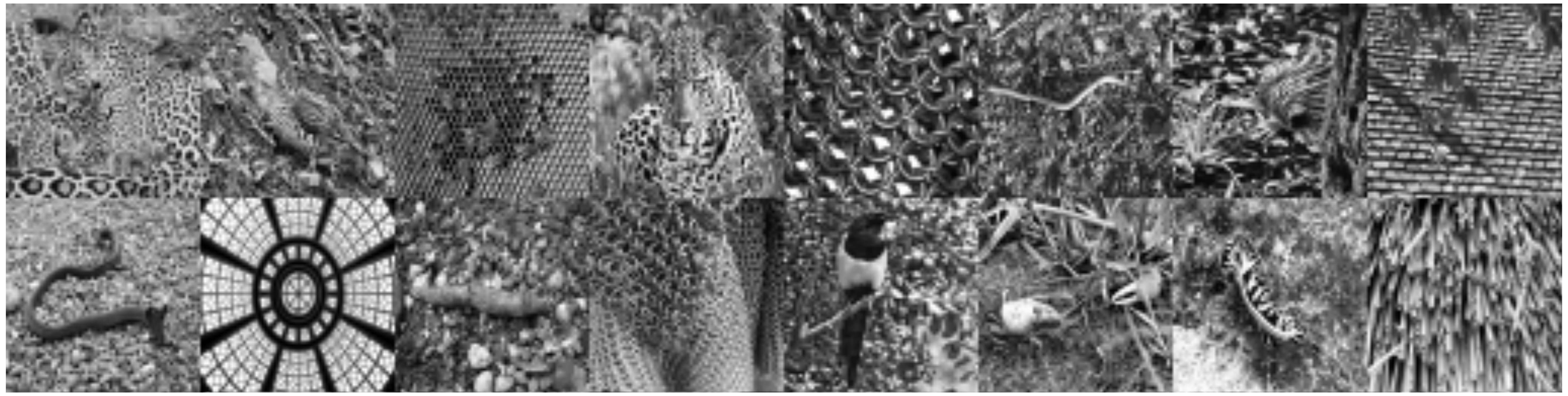
The log probability of ImageNet images

Differential entropy: -11.4 dB/dim (roughly volume of $[0, 0.1]^d$)

Quantize: out of $256^d = 10^{9,860}$ possible images, there are $10^{5,180}$ natural images

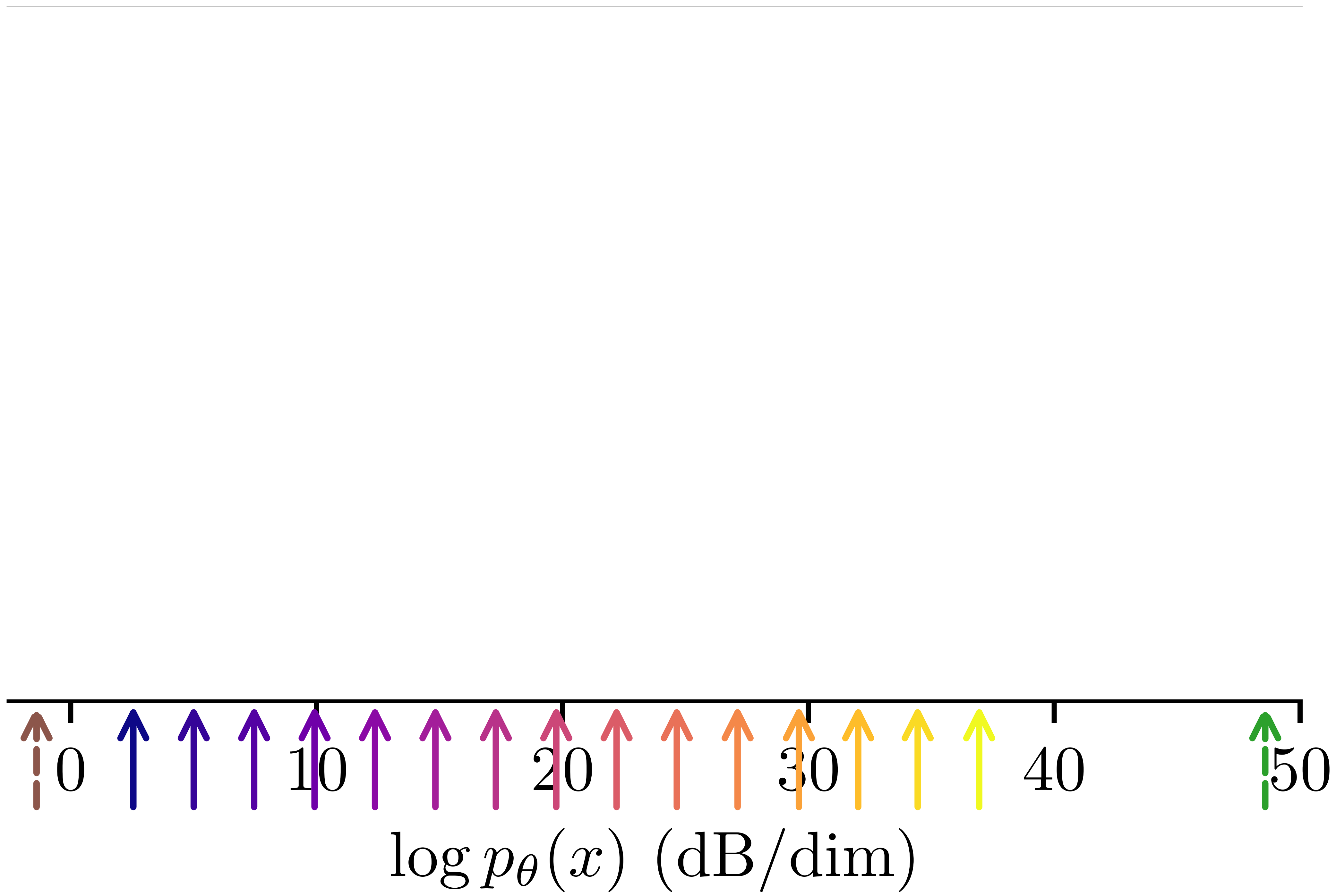
Highest probability images:

Lowest probability images:

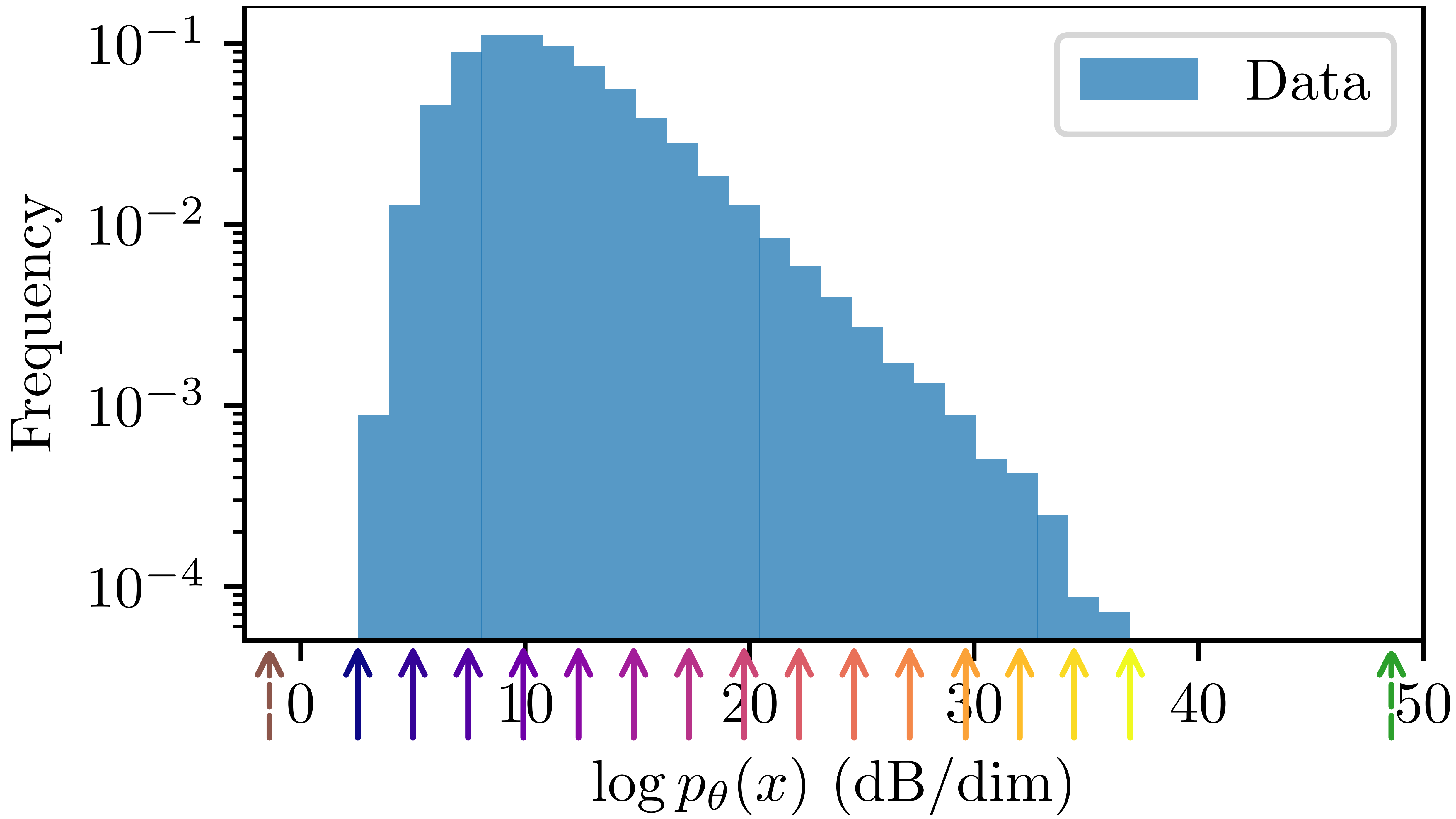


The probability ratio between these extremes is $10^{14,000}$!

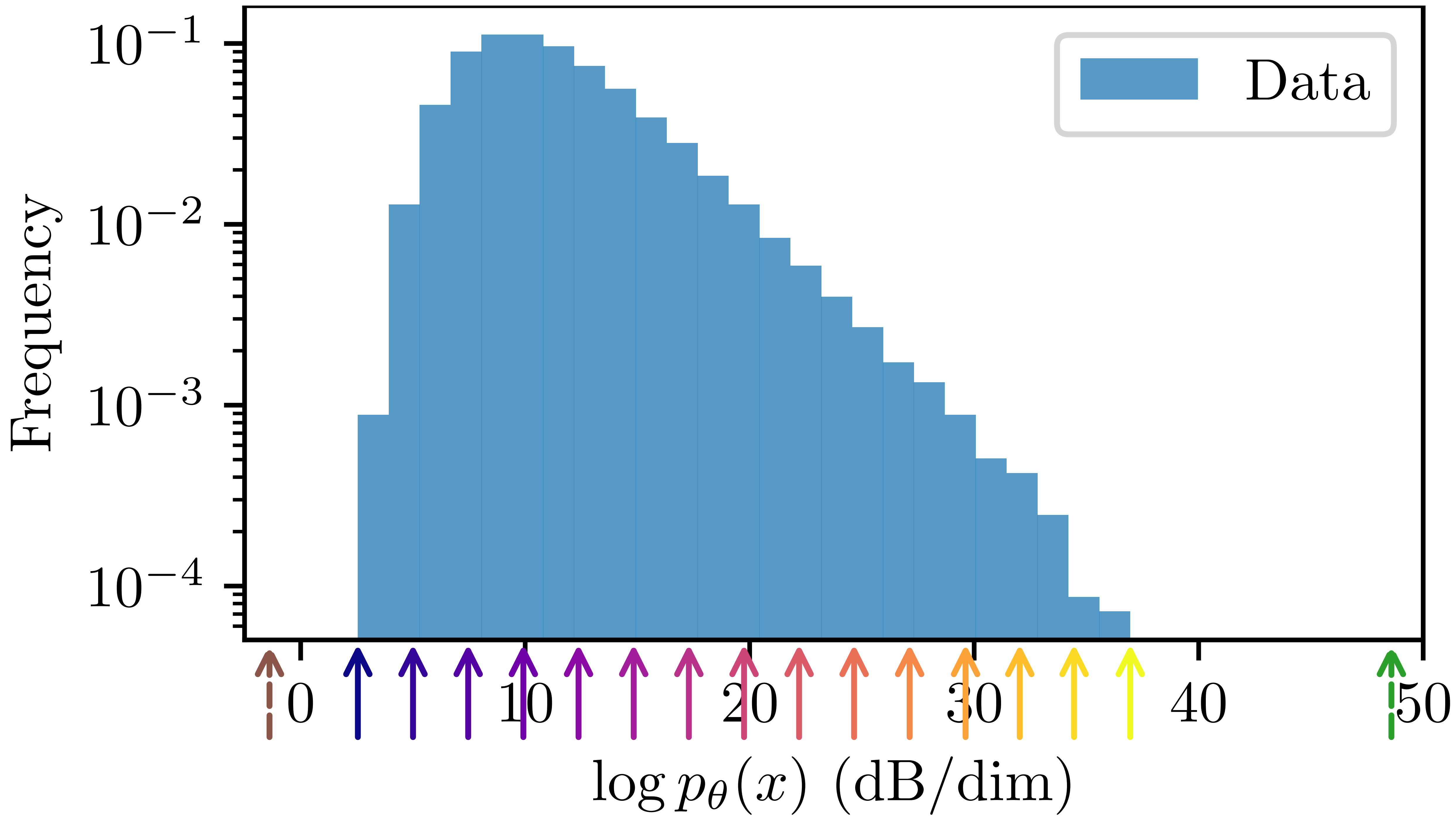
The log probability of ImageNet images



The log probability of ImageNet images

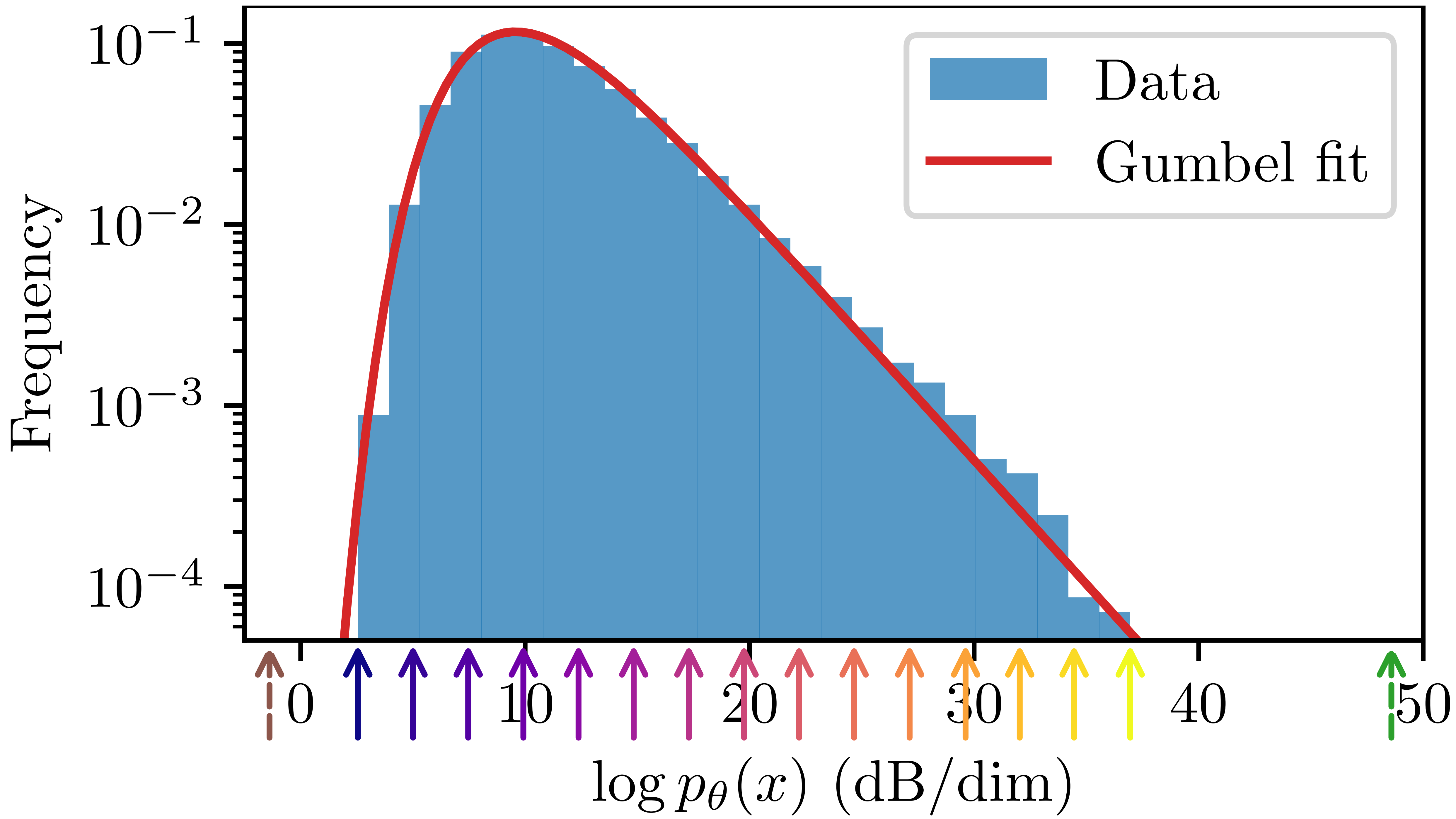


The log probability of ImageNet images



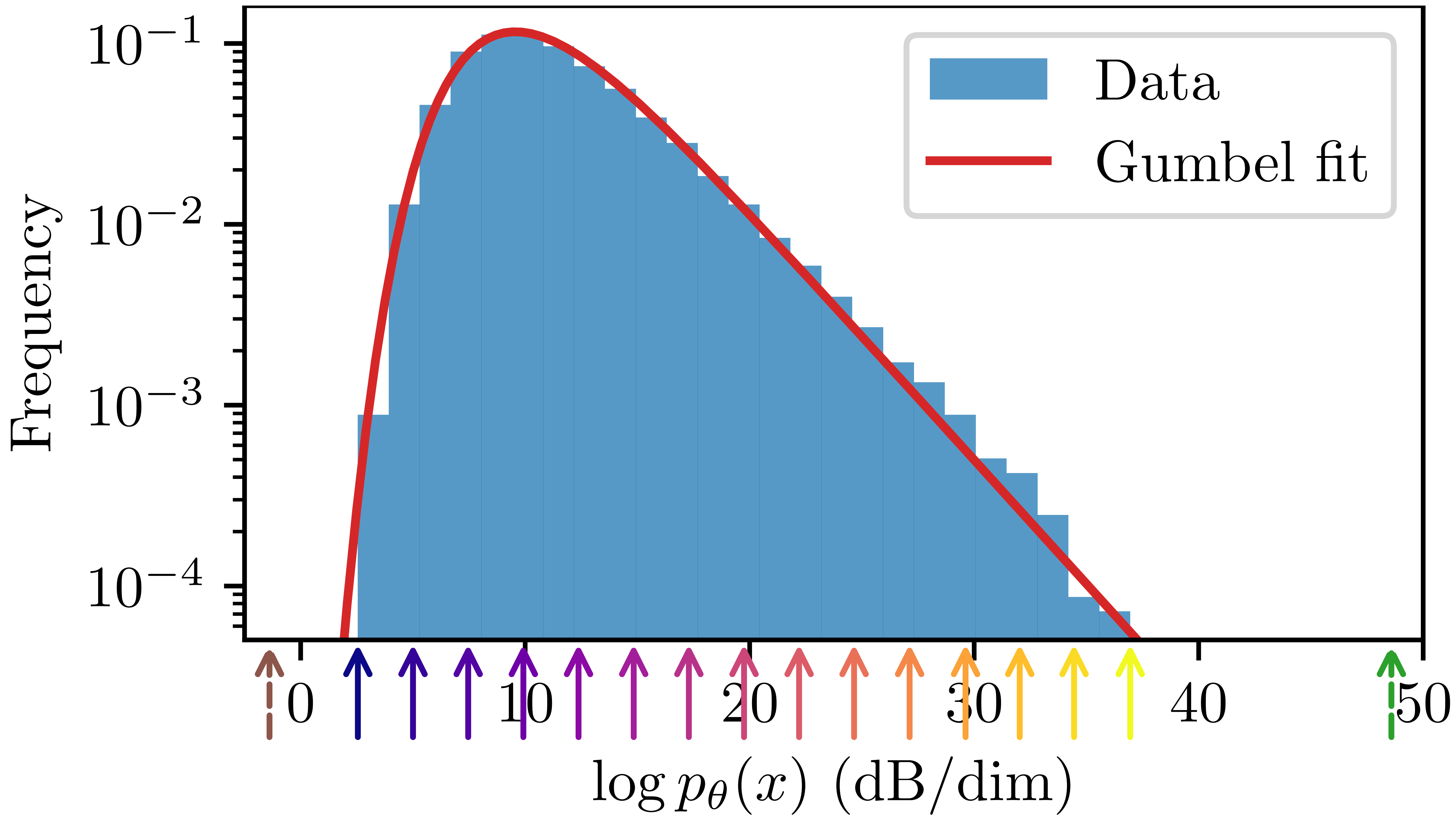
No concentration! (Volume inversely proportional to density)

The log probability of ImageNet images



No concentration! (Volume inversely proportional to density)

The log probability of ImageNet images

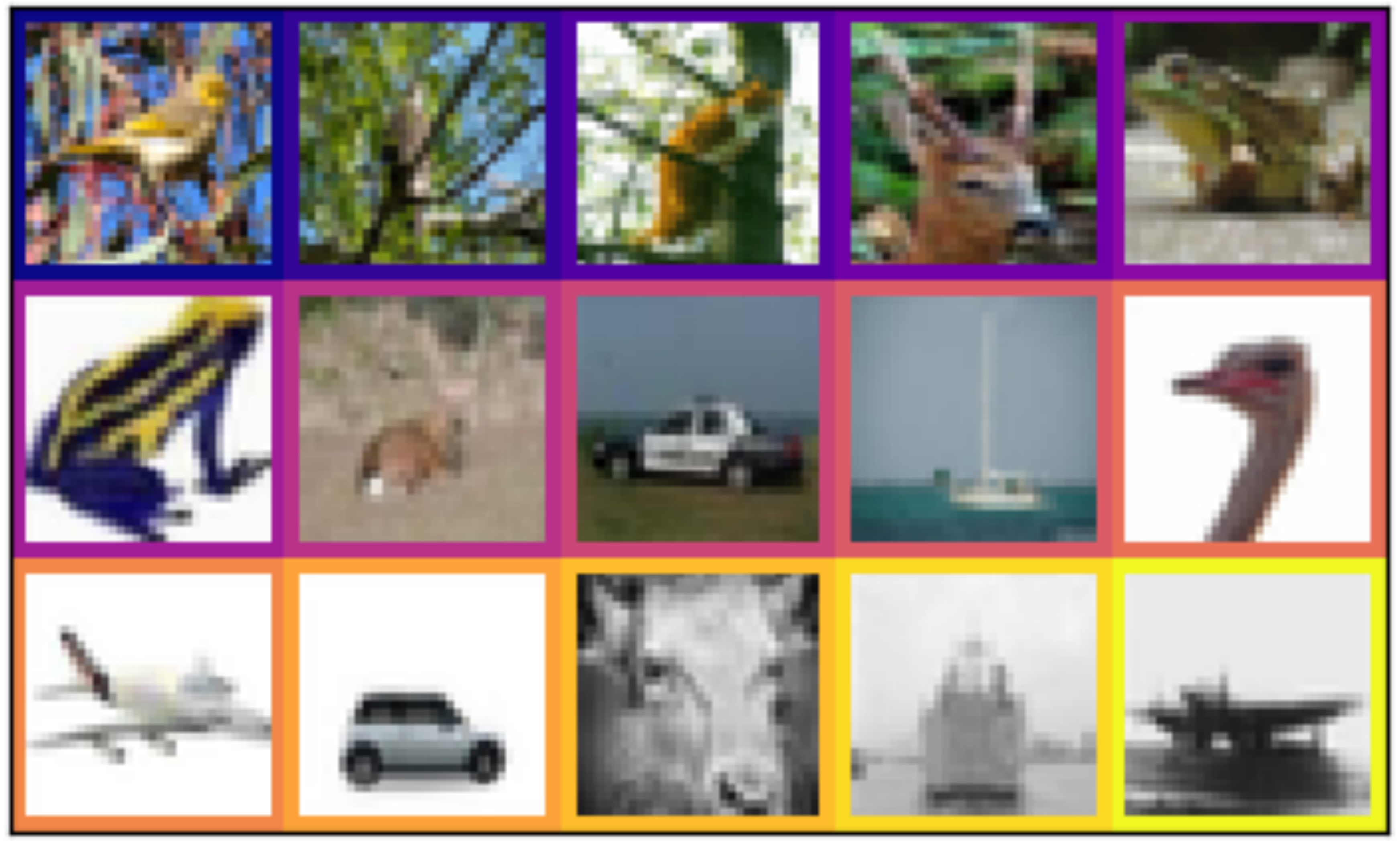


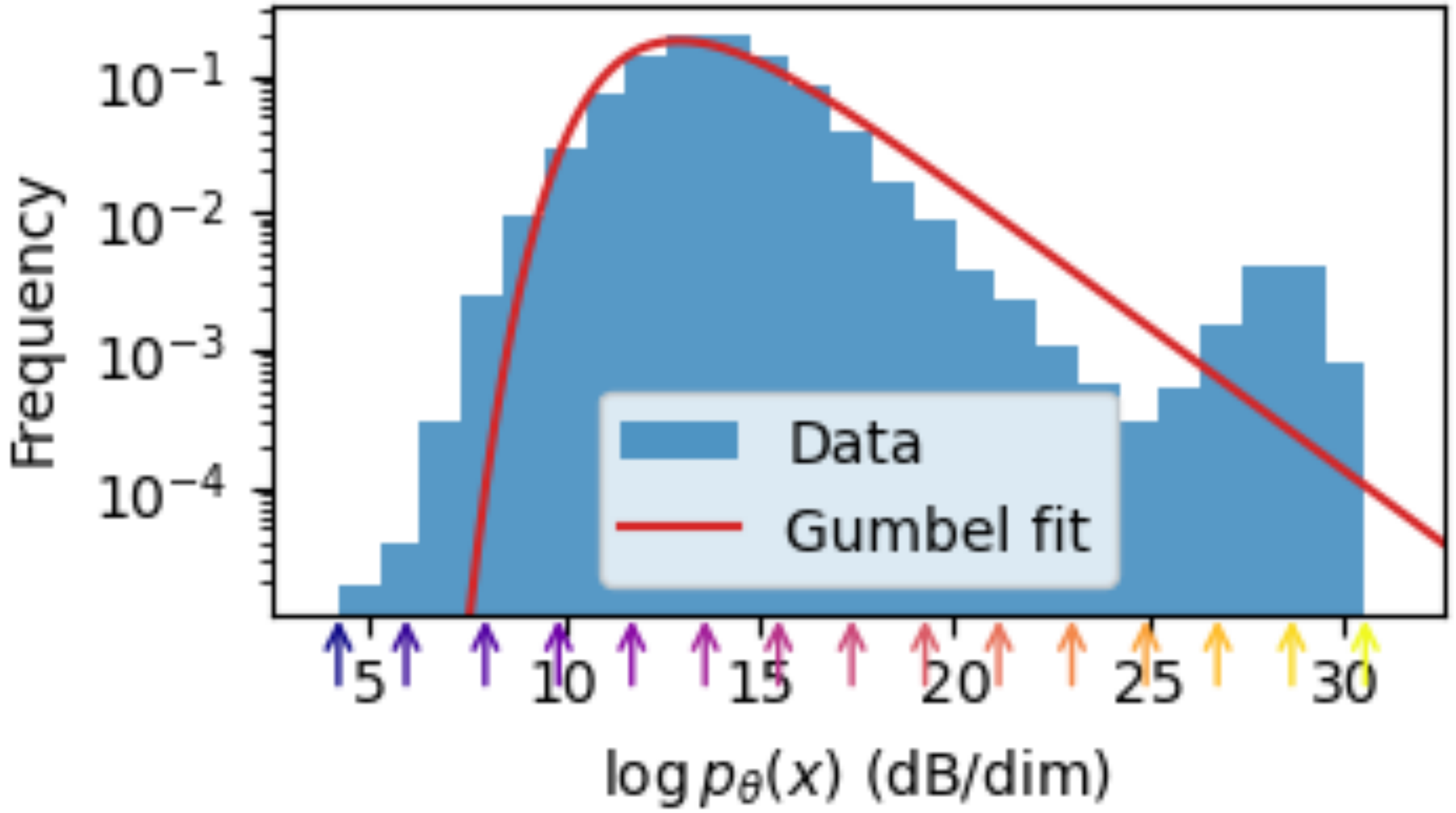
No concentration! (Volume inversely proportional to density)

Where does this Gumbel distribution come from?
(Extreme value distribution, also appears in Gaussian scale mixtures)

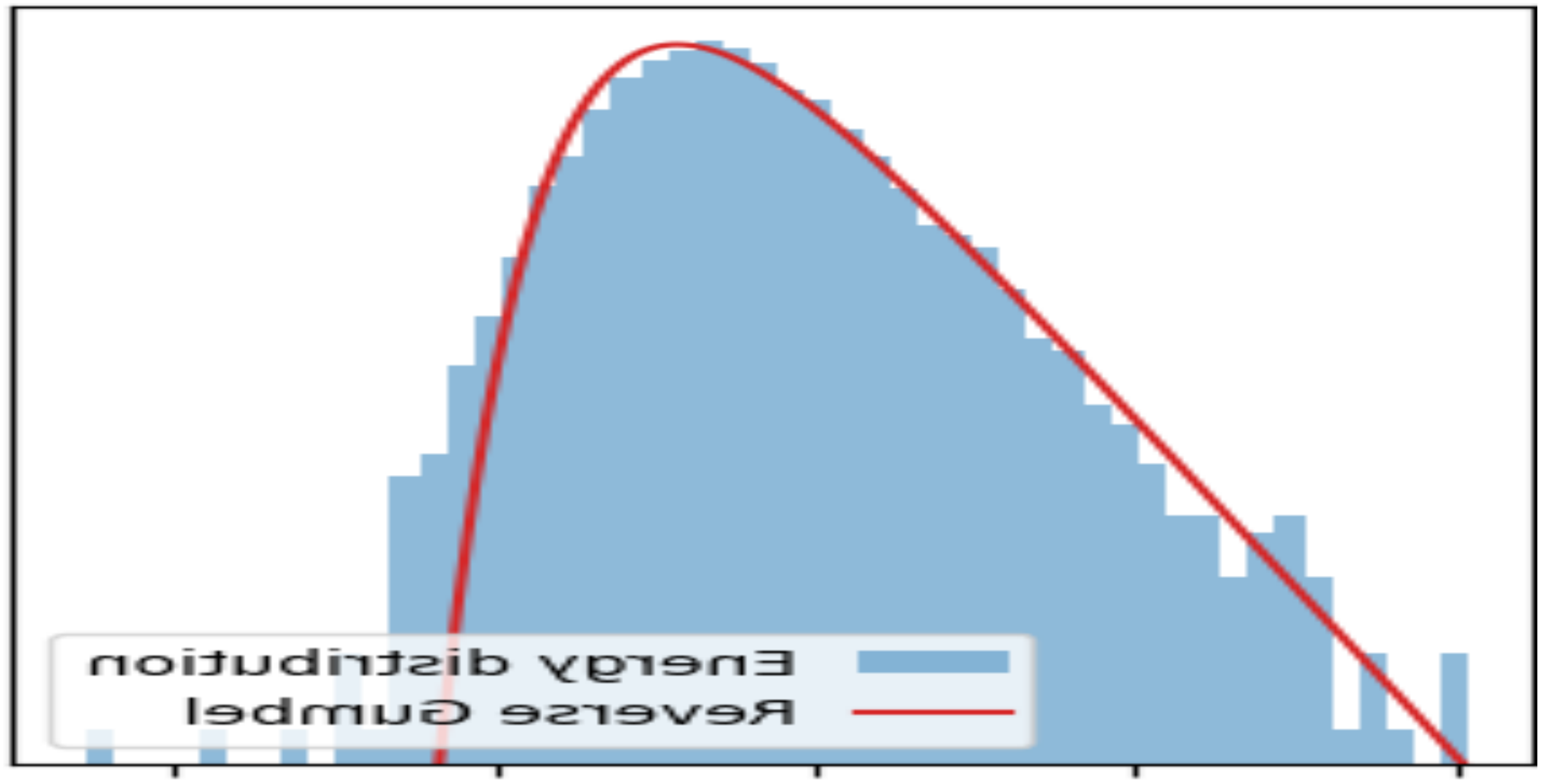
Gumbel fits on other datasets

CIFAR-10:

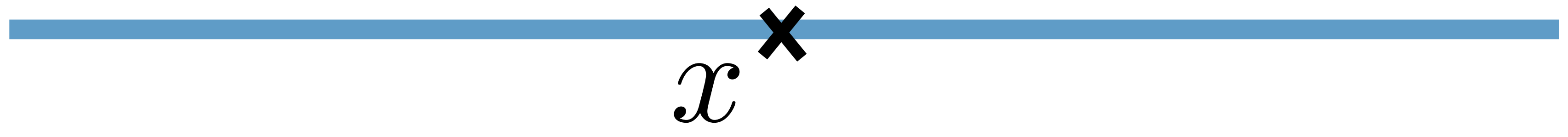




CelebA:

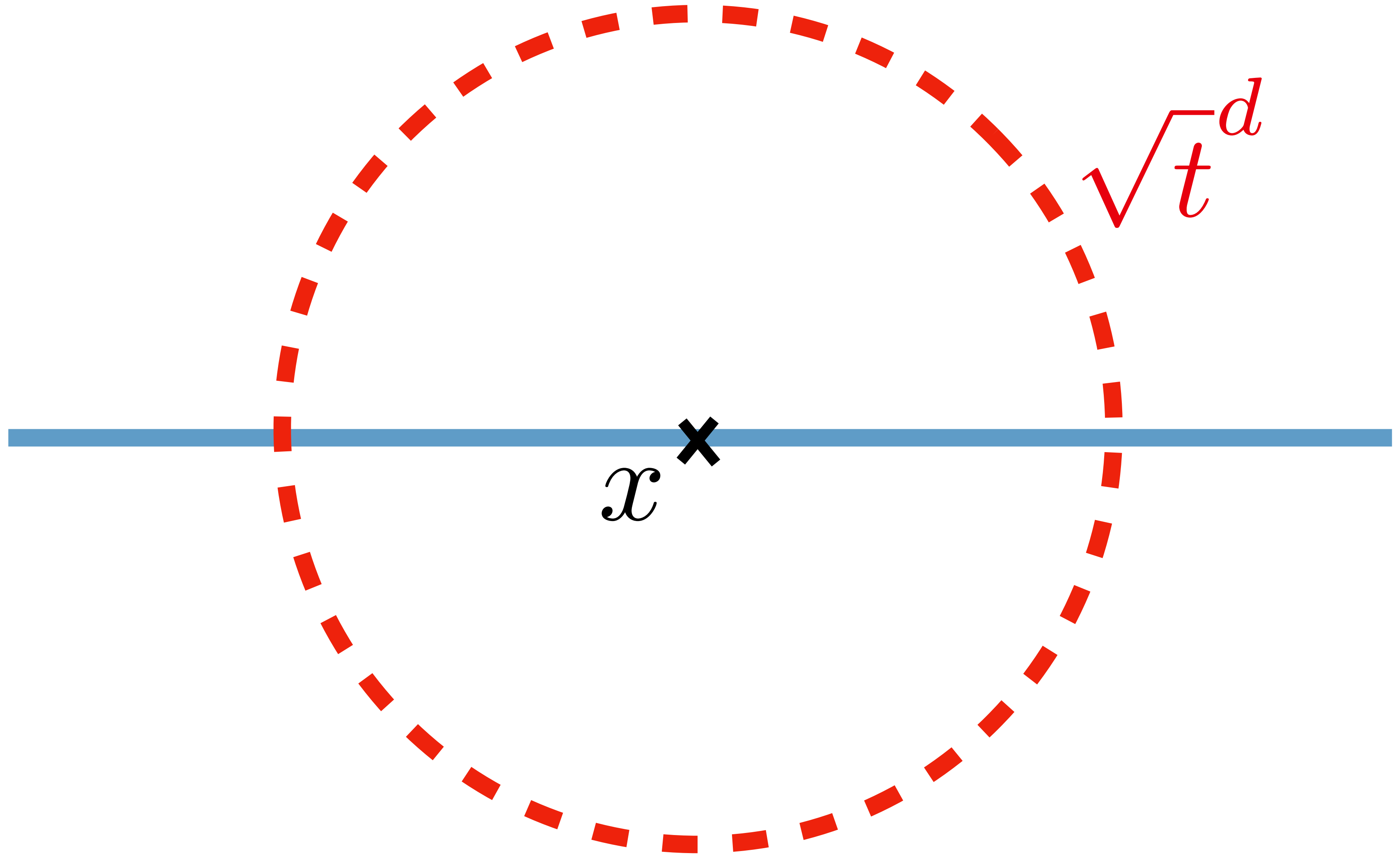


The local behavior of log probability



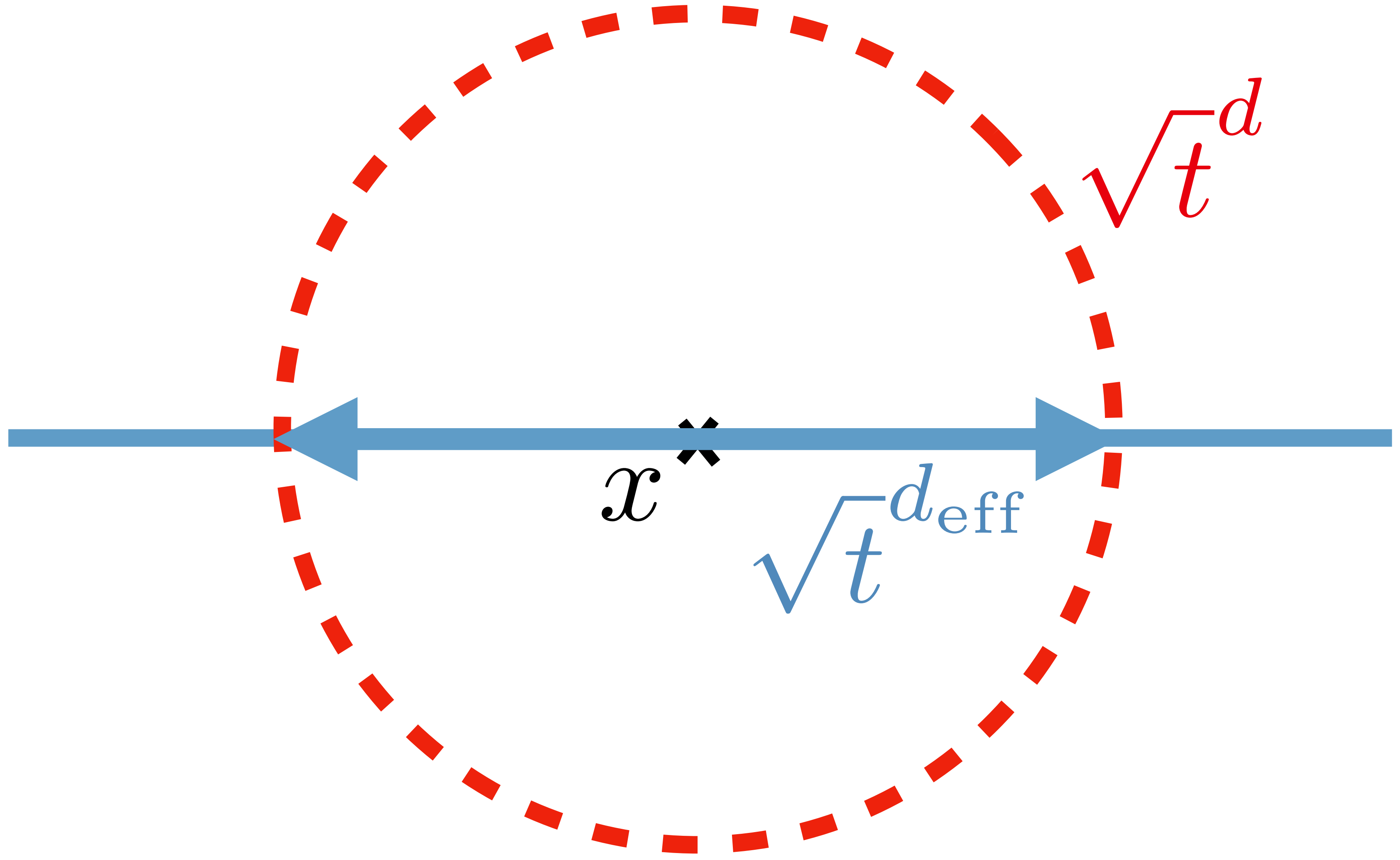
(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

The local behavior of log probability



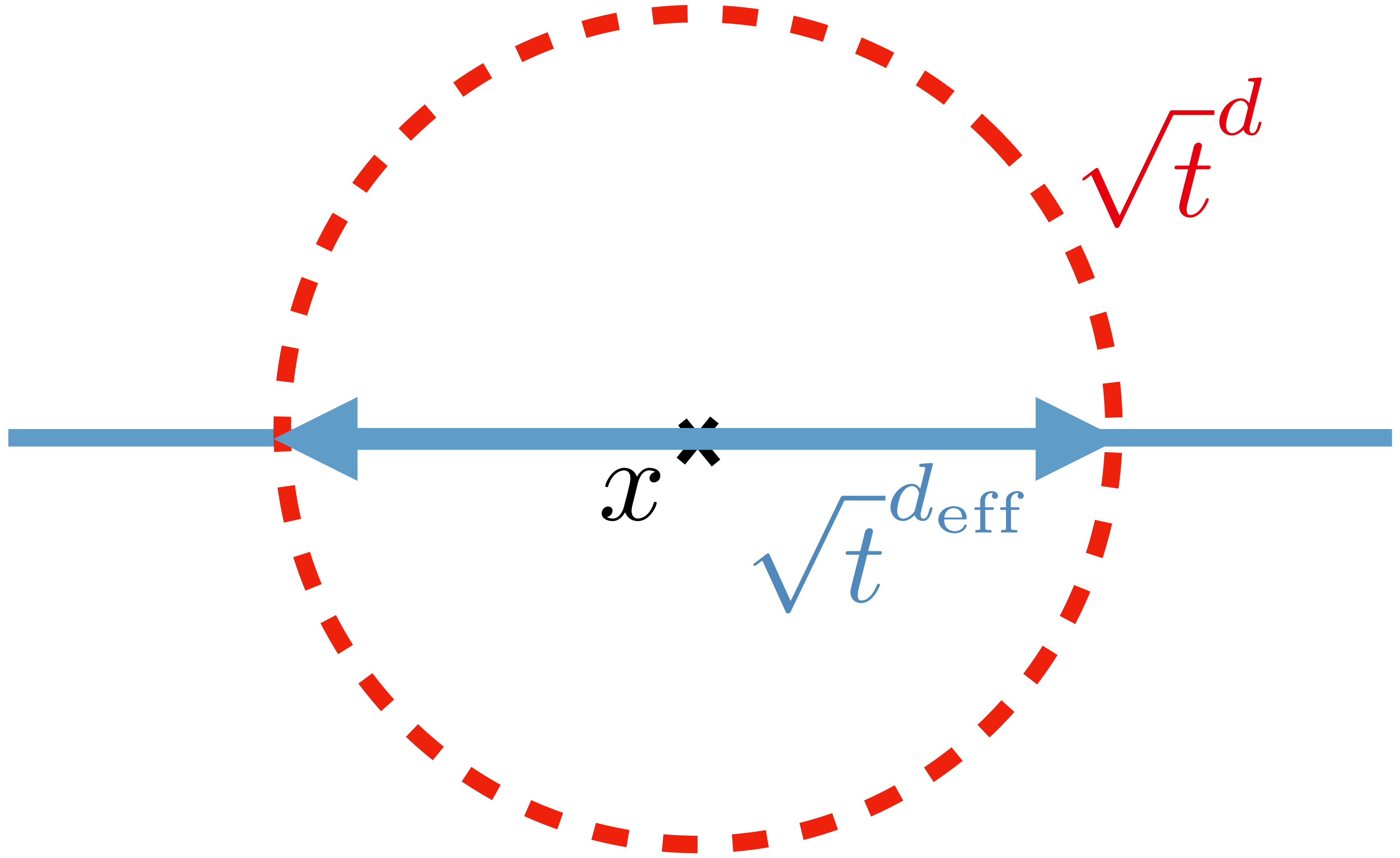
(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

The local behavior of log probability



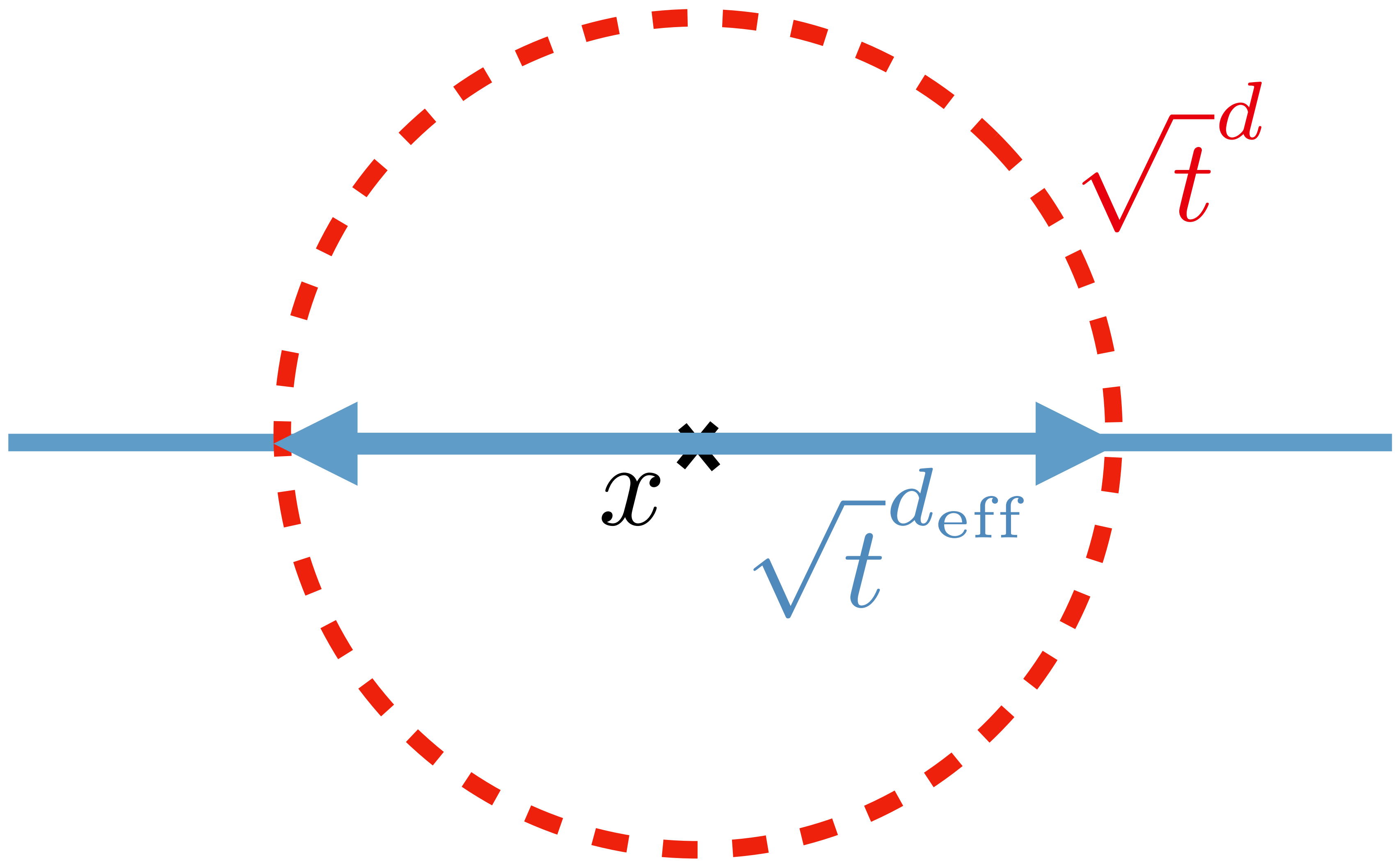
(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

The local behavior of log probability



(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

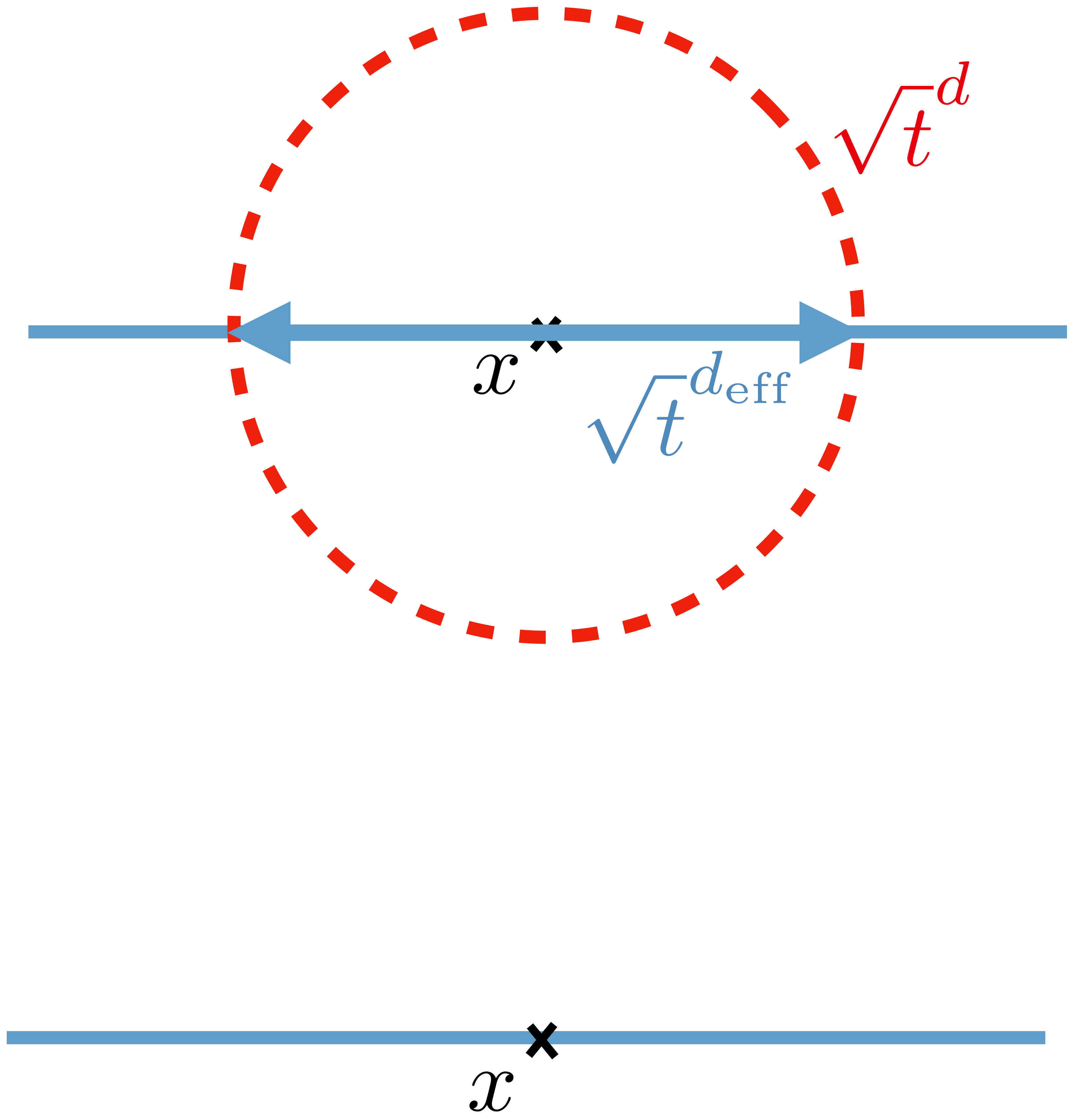
The local behavior of log probability



$$d_{\text{eff}}(x,t) = d + 2t\partial_t \mathbb{E}_y[\log p_\theta(y,t) \mid x]$$

(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

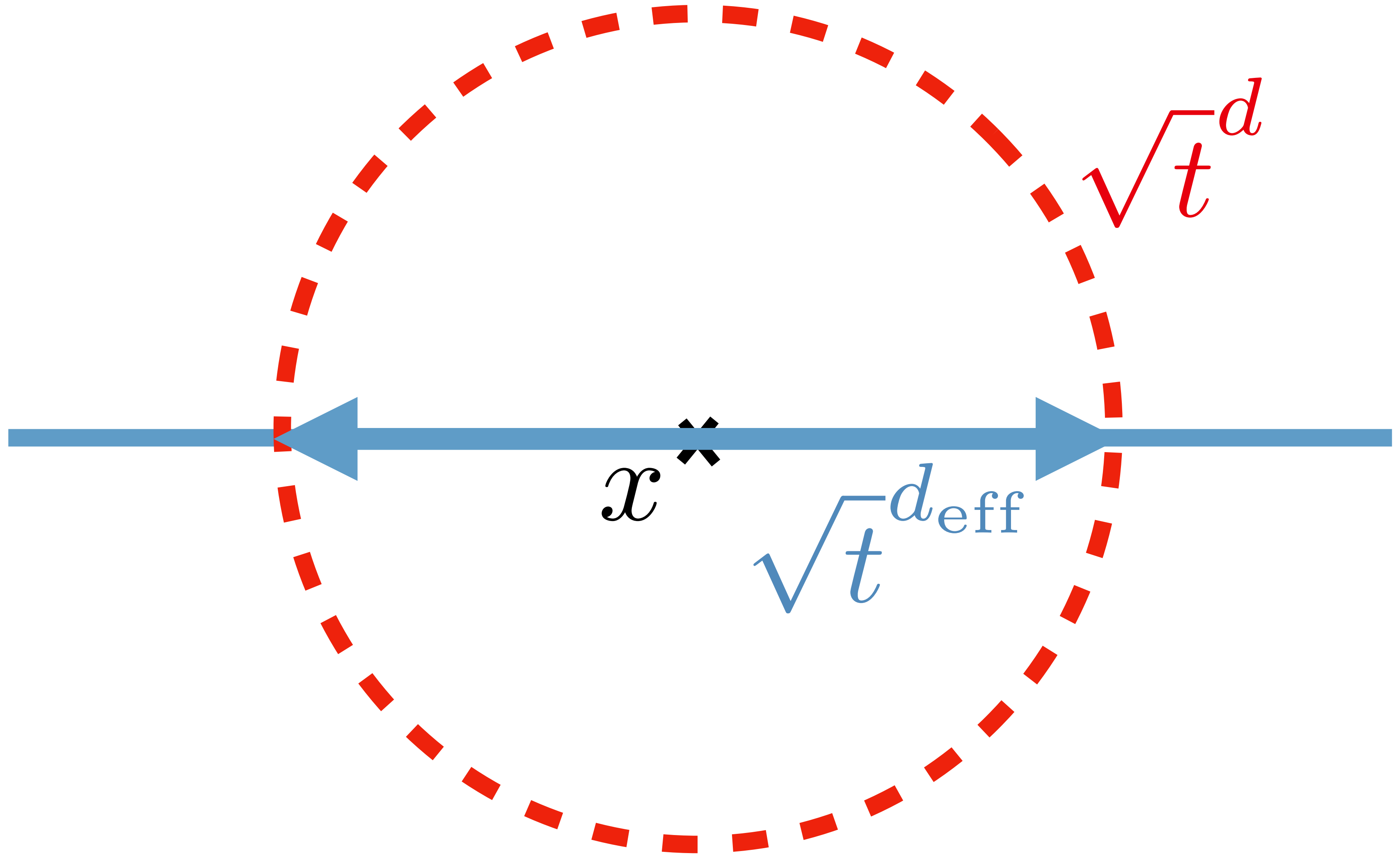
The local behavior of log probability



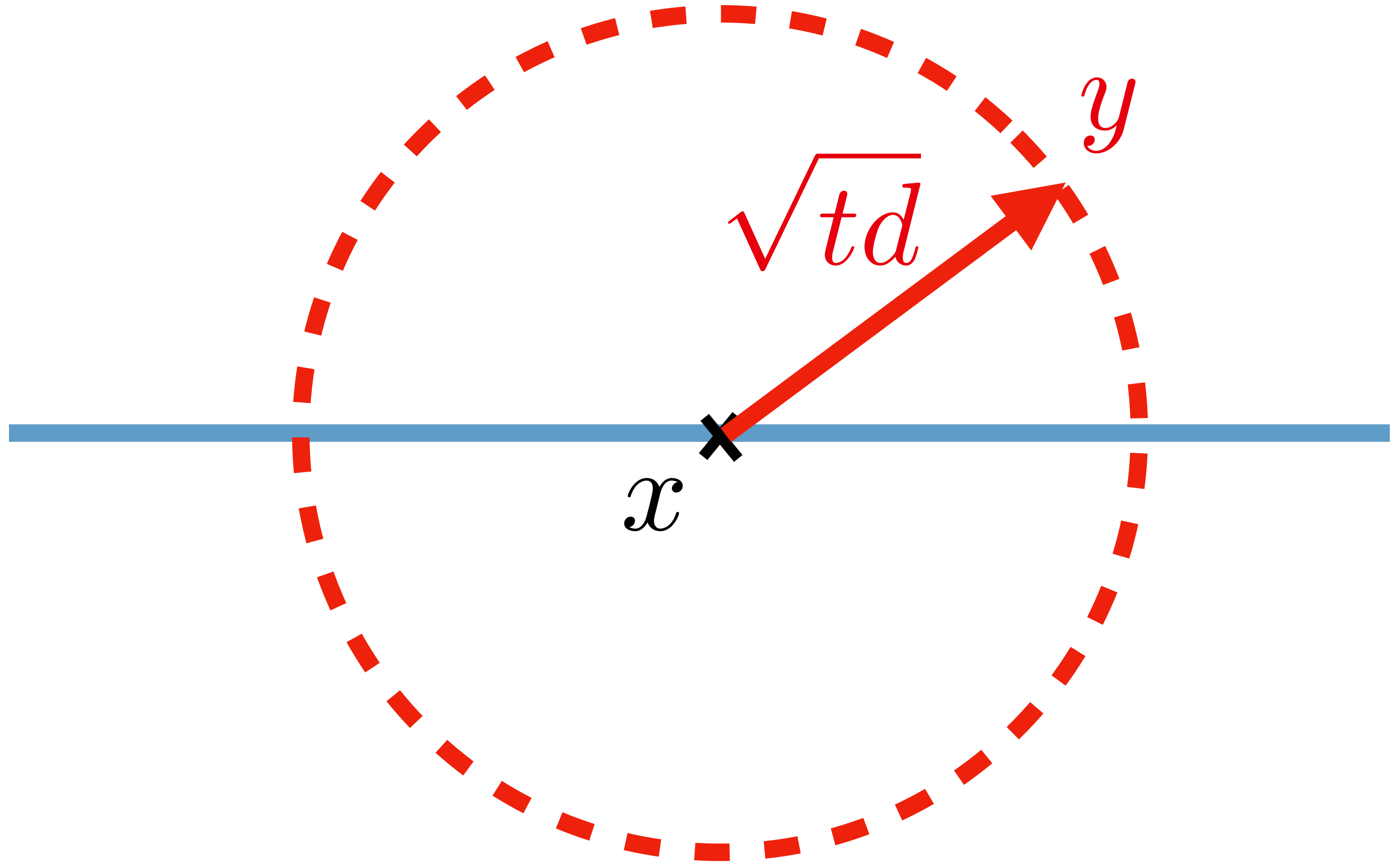
$$d_{\text{eff}}(x,t) = d + 2t\partial_t \mathbb{E}_y[\log p_\theta(y,t)\,|\,x]$$

(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

The local behavior of log probability

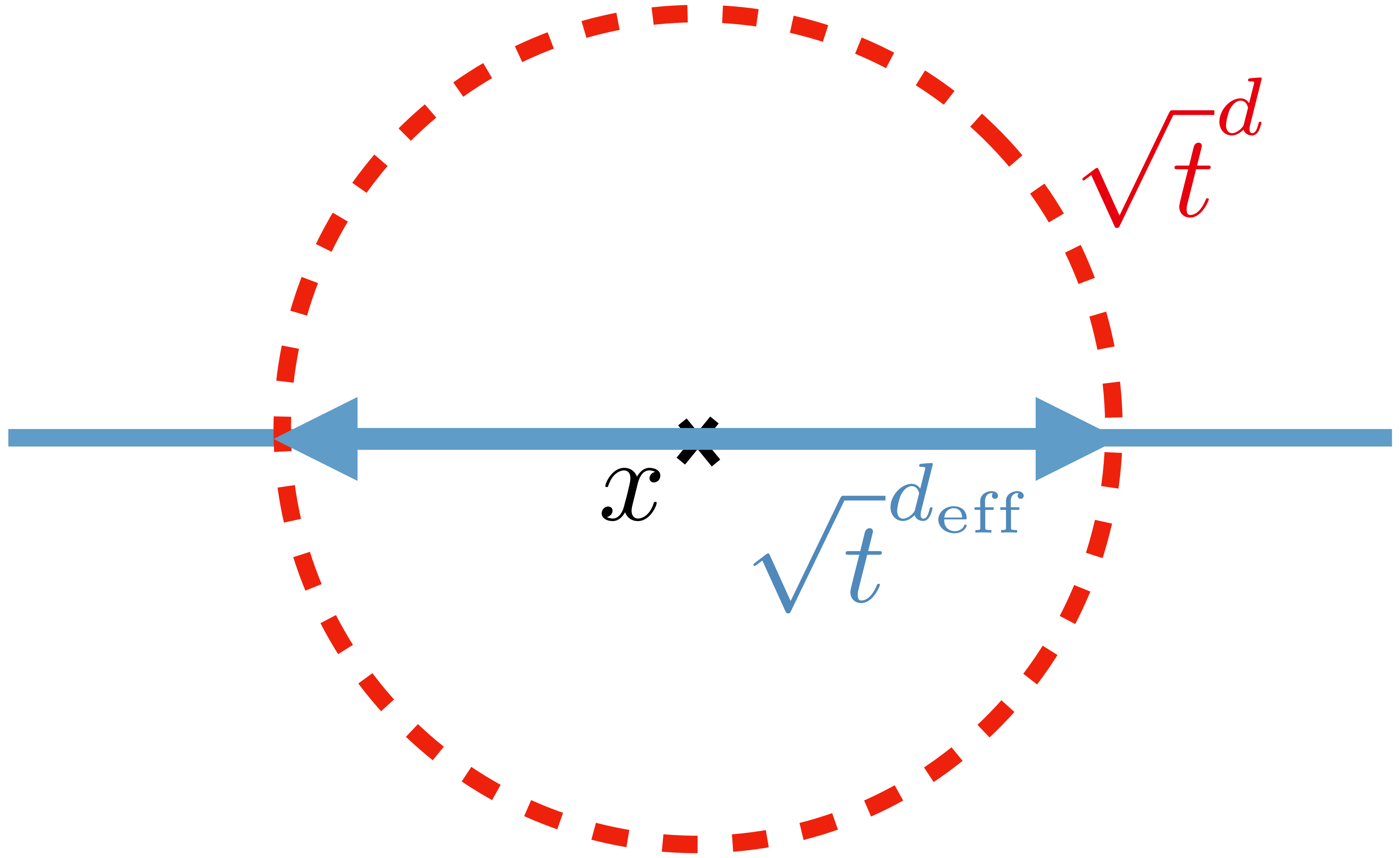


$$d_{\text{eff}}(x,t) = d + 2t\partial_t \mathbb{E}_y[\log p_\theta(y,t) \mid x]$$

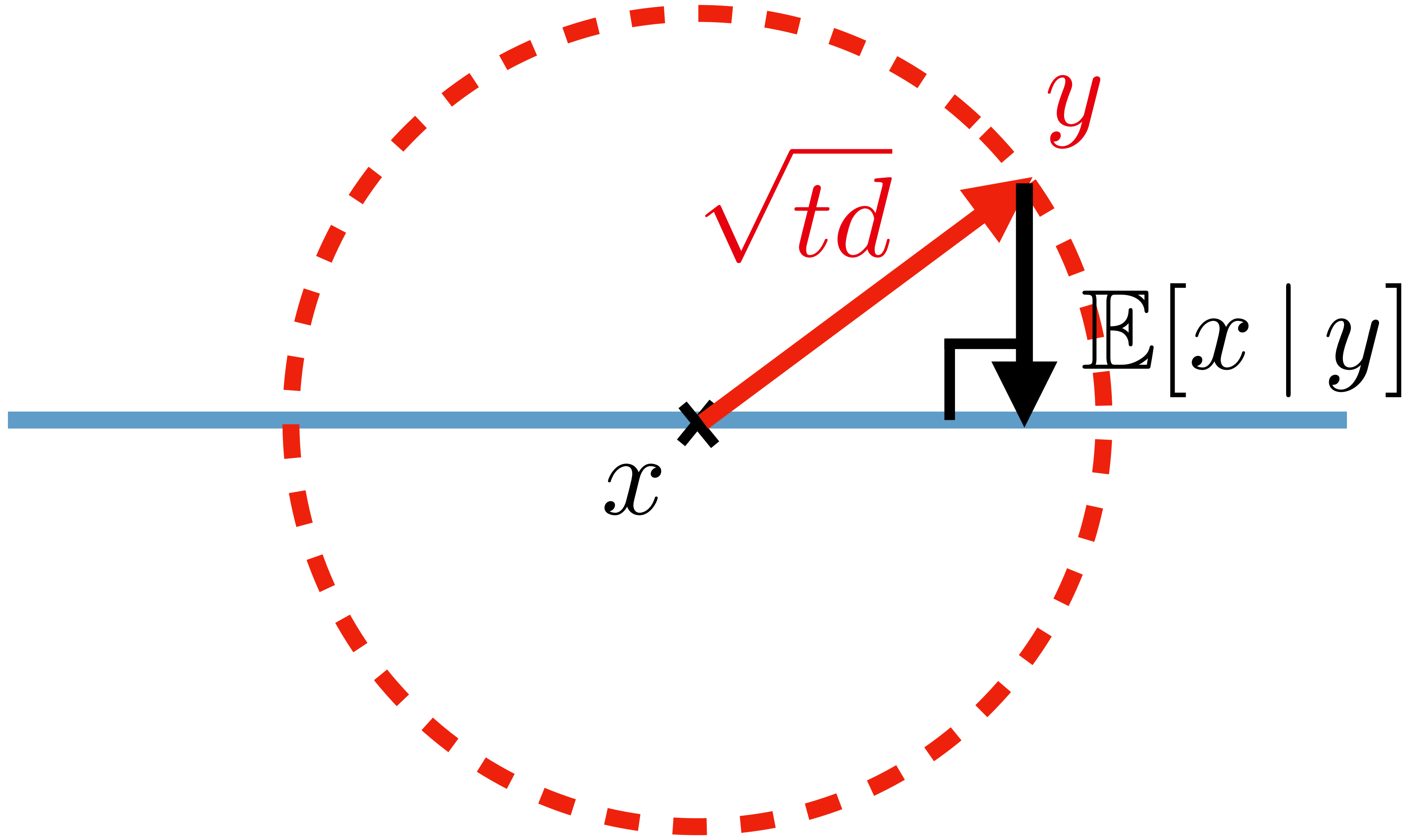


(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

The local behavior of log probability

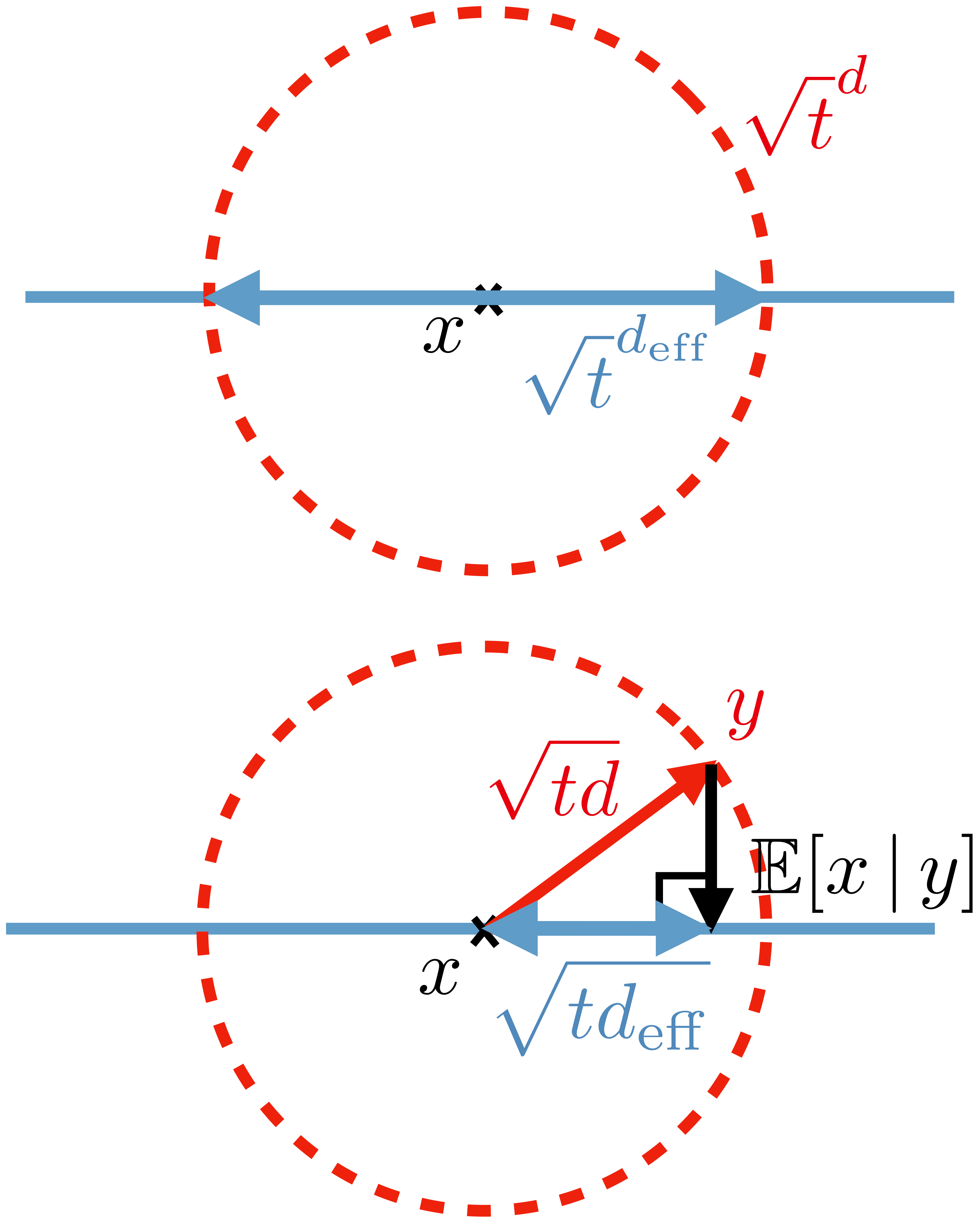


$$d_{\text{eff}}(x,t) = d + 2t\partial_t \mathbb{E}_y[\log p_\theta(y,t) \mid x]$$



(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

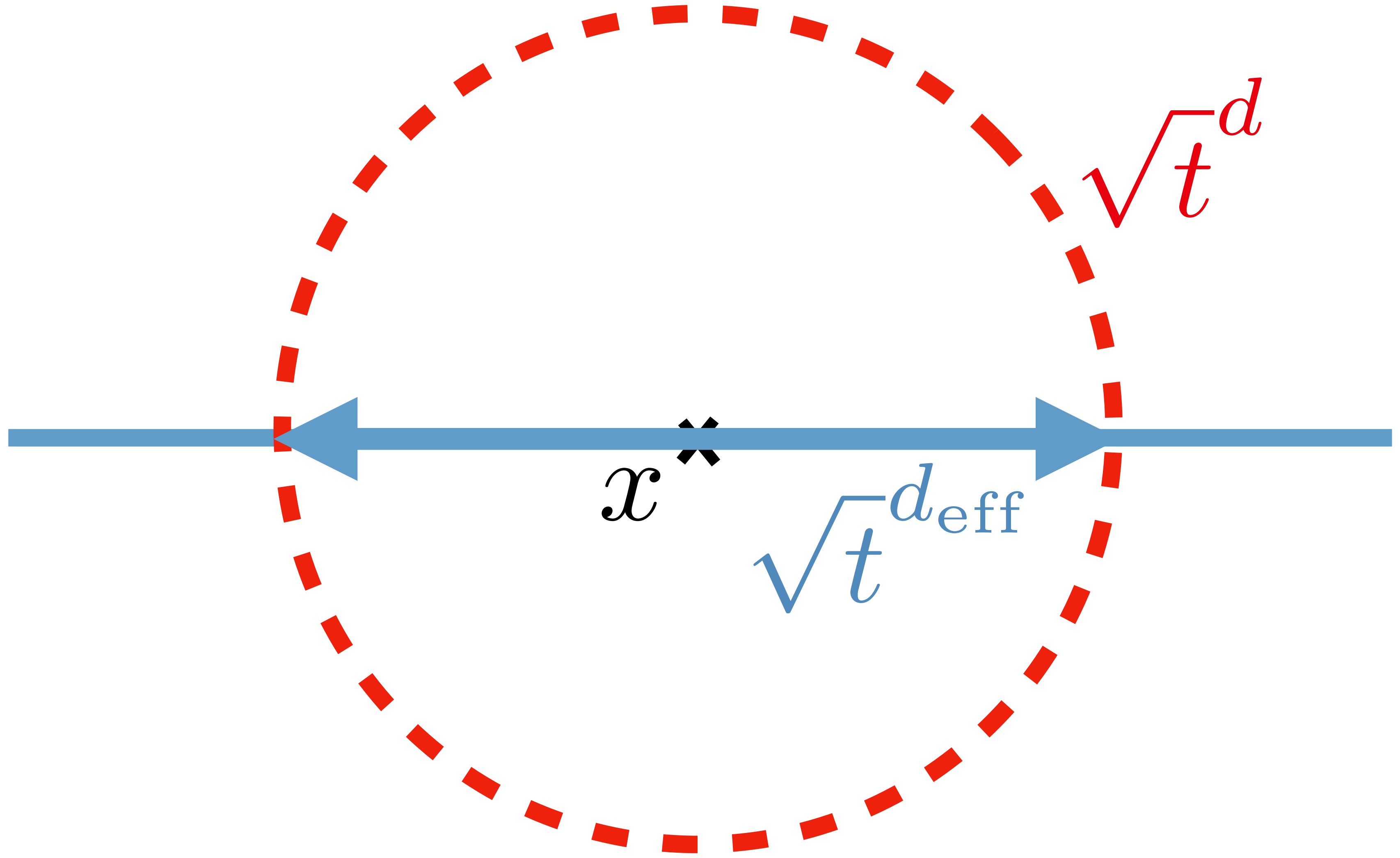
The local behavior of log probability



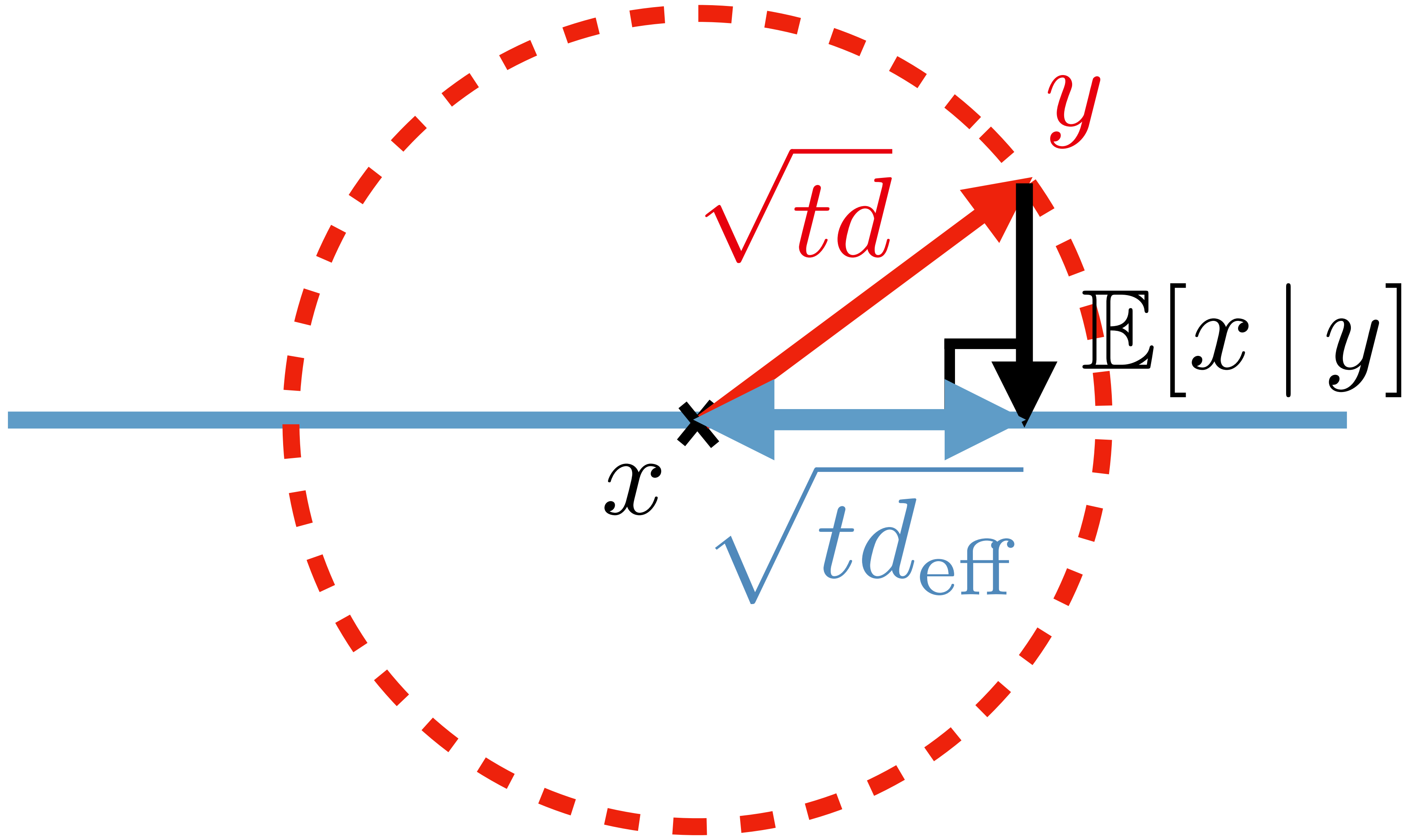
$$d_{\text{eff}}(x,t) = d + 2t\partial_t \mathbb{E}_y[\log p_\theta(y,t) \mid x]$$

(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

The local behavior of log probability



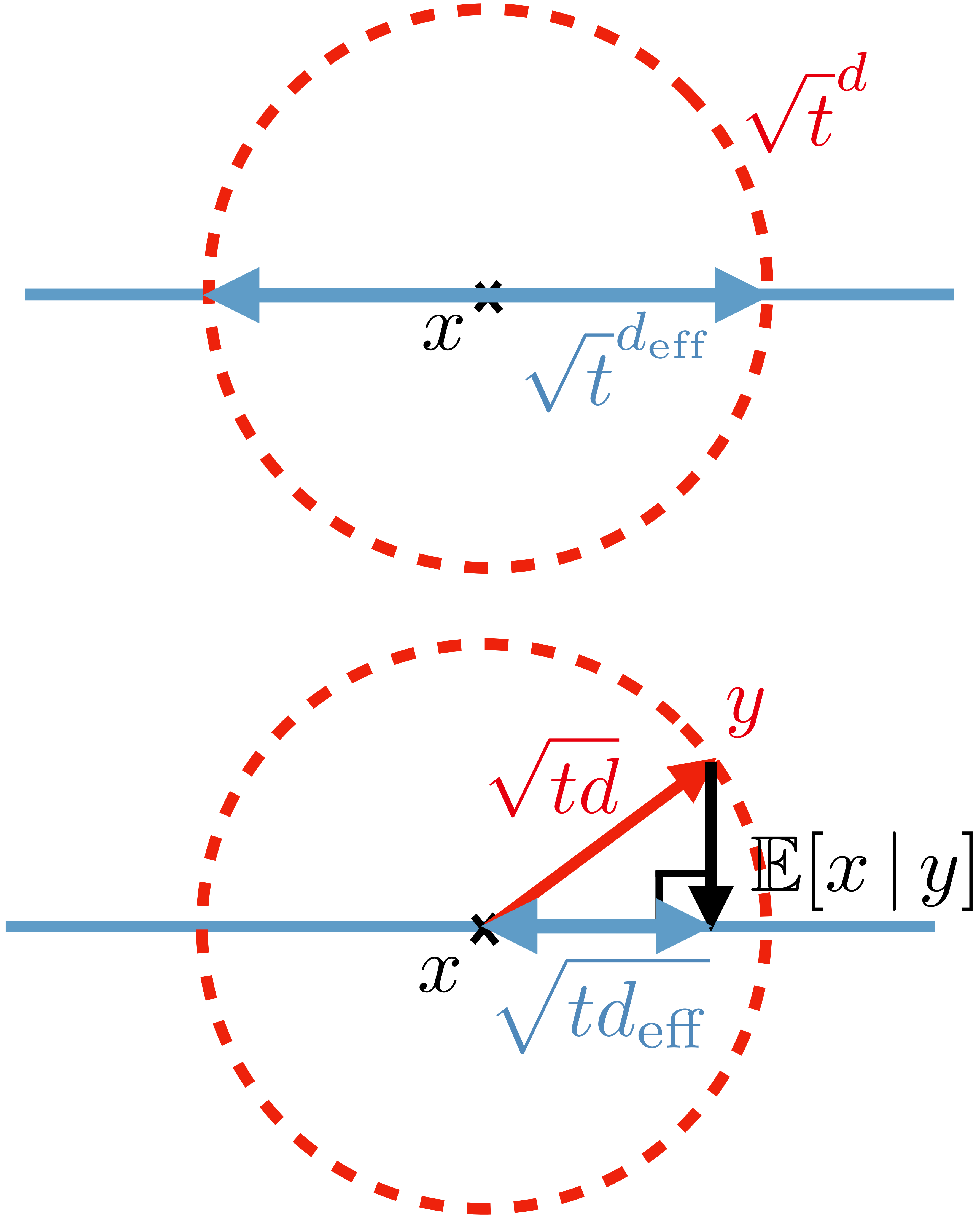
$$d_{\text{eff}}(x,t) = d + 2t\partial_t \mathbb{E}_y[\log p_\theta(y,t) \mid x]$$



$$d_{\text{eff}}(x,t) = \frac{1}{t}\mathbb{E}_y[\|x - \mathbb{E}_x[x \mid y]\|^2 \mid x]$$

(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

The local behavior of log probability



$$d_{\text{eff}}(x, t) = d + 2t\partial_t \mathbb{E}_y[\log p_\theta(y, t) \mid x]$$

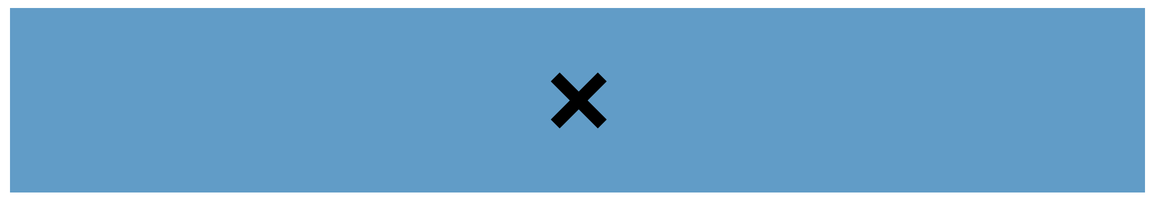
These two definitions coincide!

$$d_{\text{eff}}(x, t) = \frac{1}{t}\mathbb{E}_y[\|x - \mathbb{E}_x[x \mid y]\|^2 \mid x]$$

(Wu and Verdú, 2011; Mohan*, Kadkhodaie* et al., 2019; Tempczyk et al., 2022; Stanczuk et al., 2022; Kamkari et al., 2024)

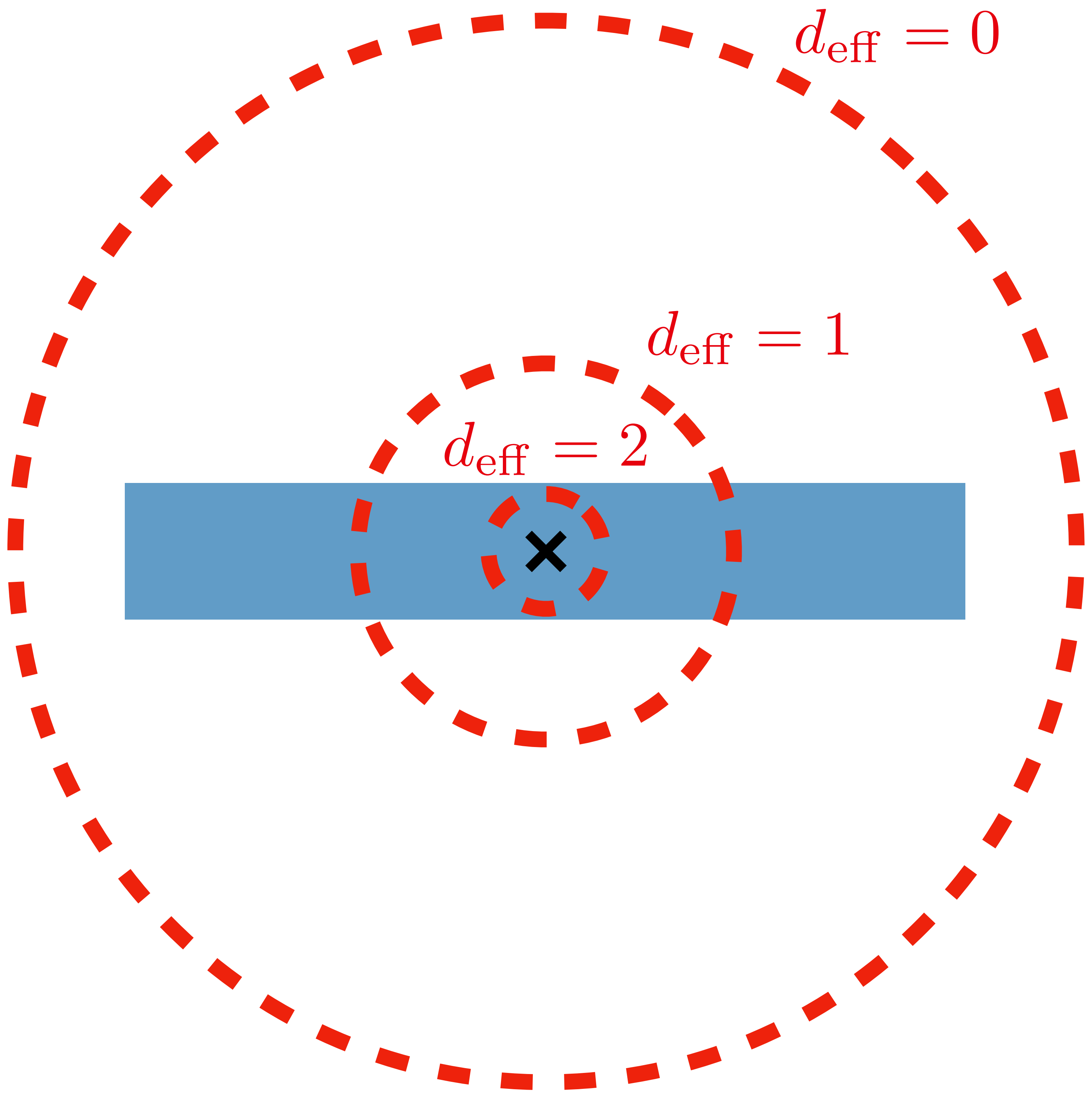
Testing the manifold hypothesis

Dimensionality is a scale-dependent-measure!



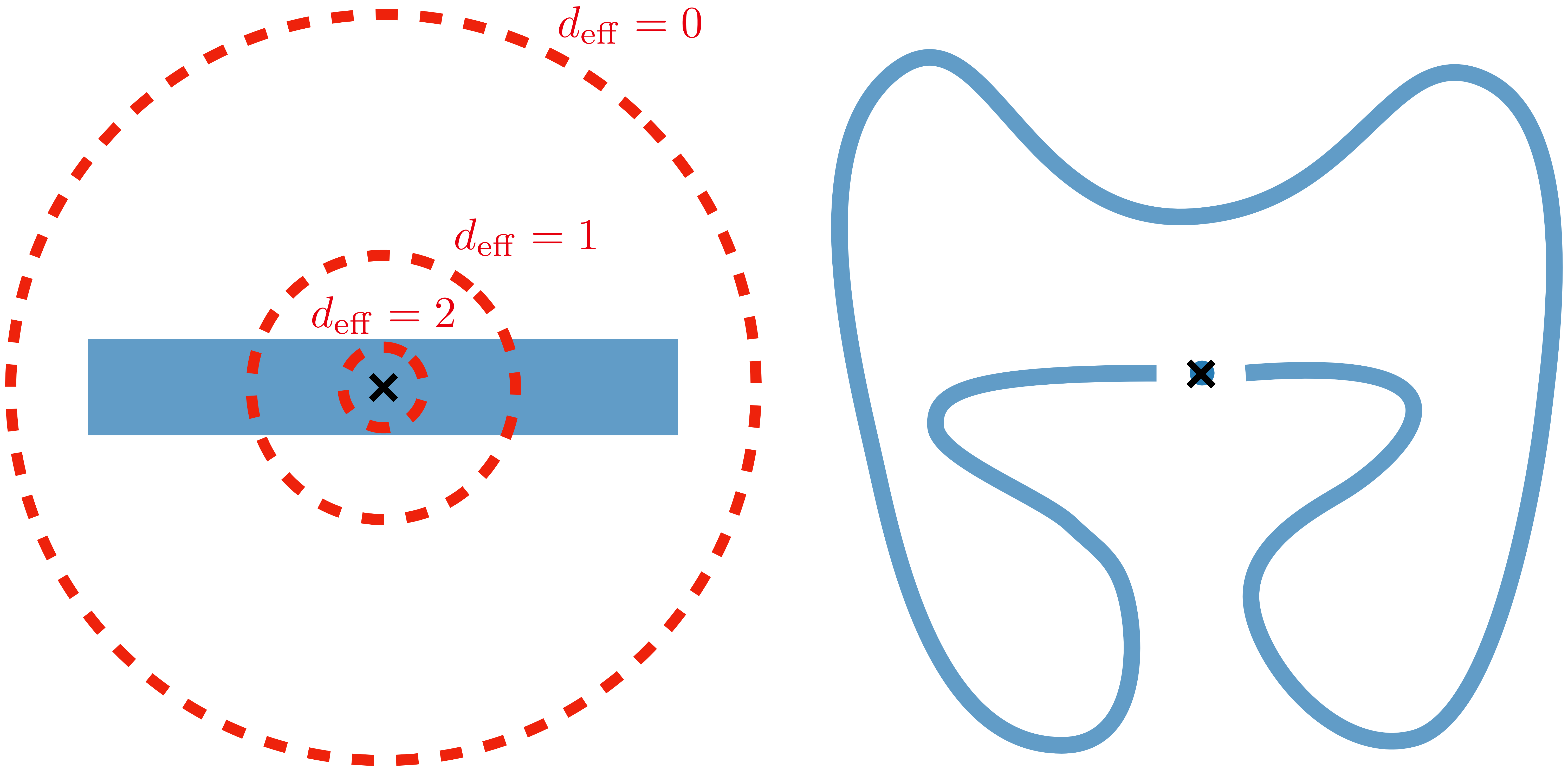
Testing the manifold hypothesis

Dimensionality is a scale-dependent-measure!



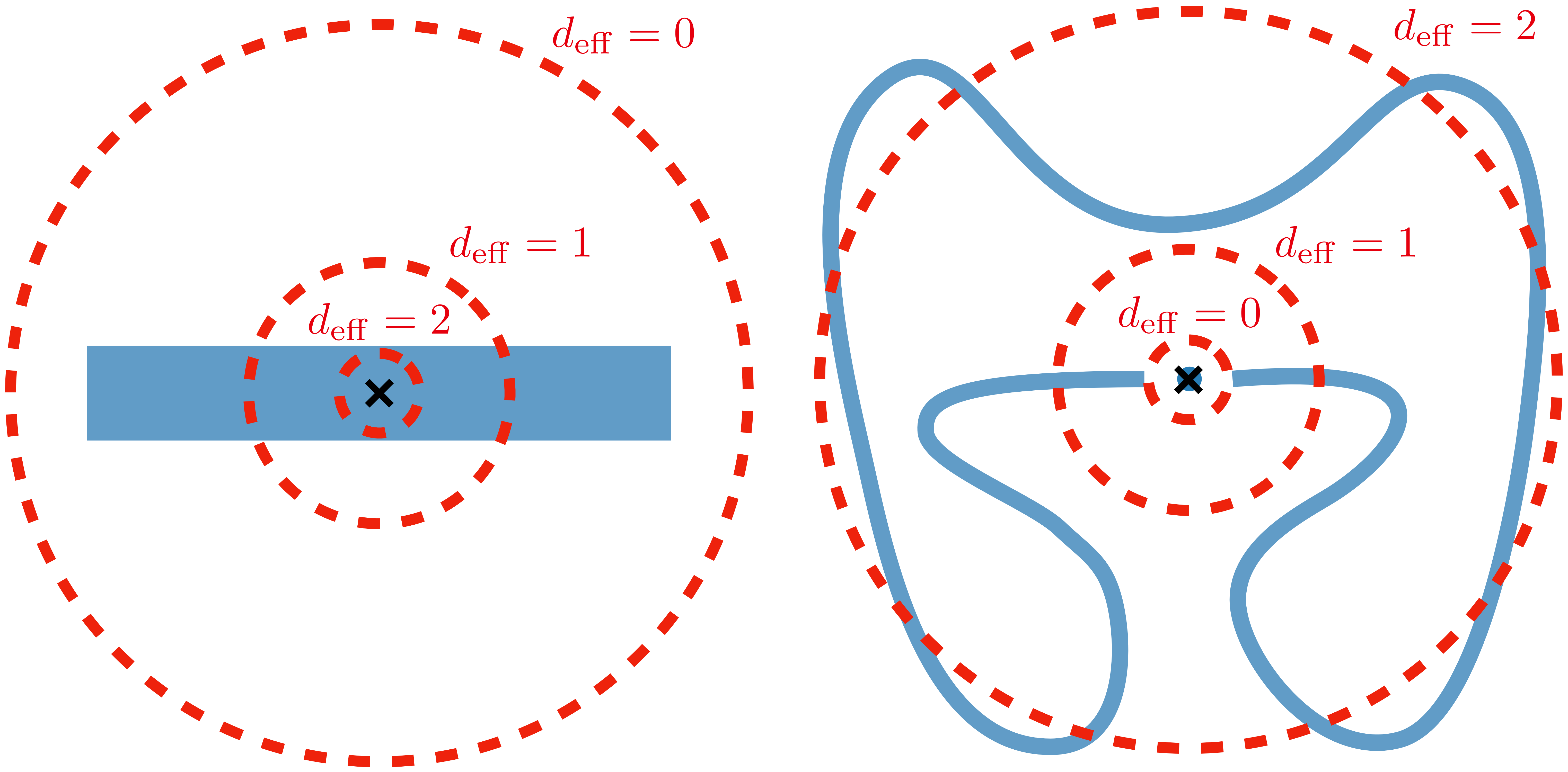
Testing the manifold hypothesis

Dimensionality is a scale-dependent-measure!



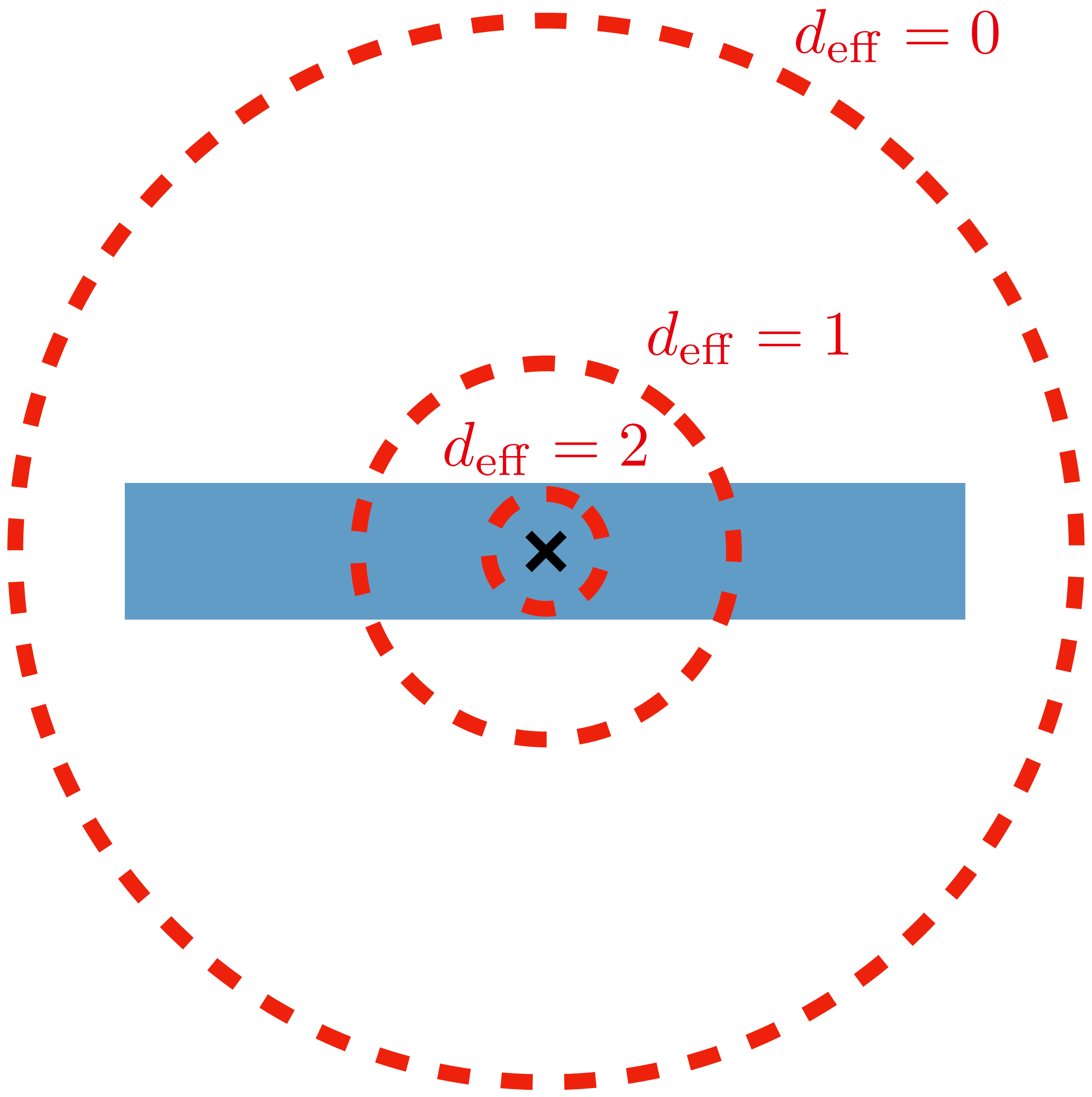
Testing the manifold hypothesis

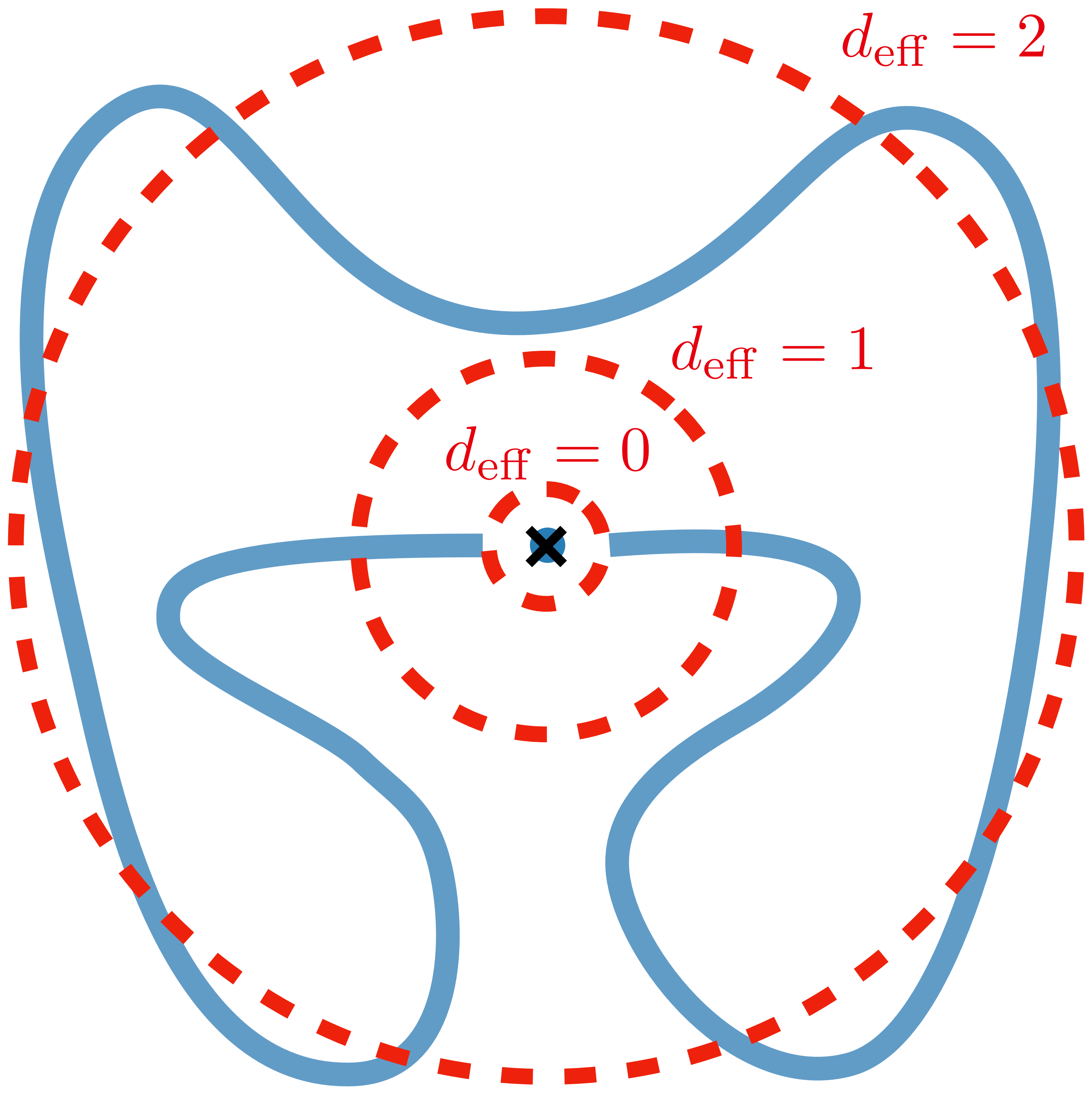
Dimensionality is a scale-dependent-measure!

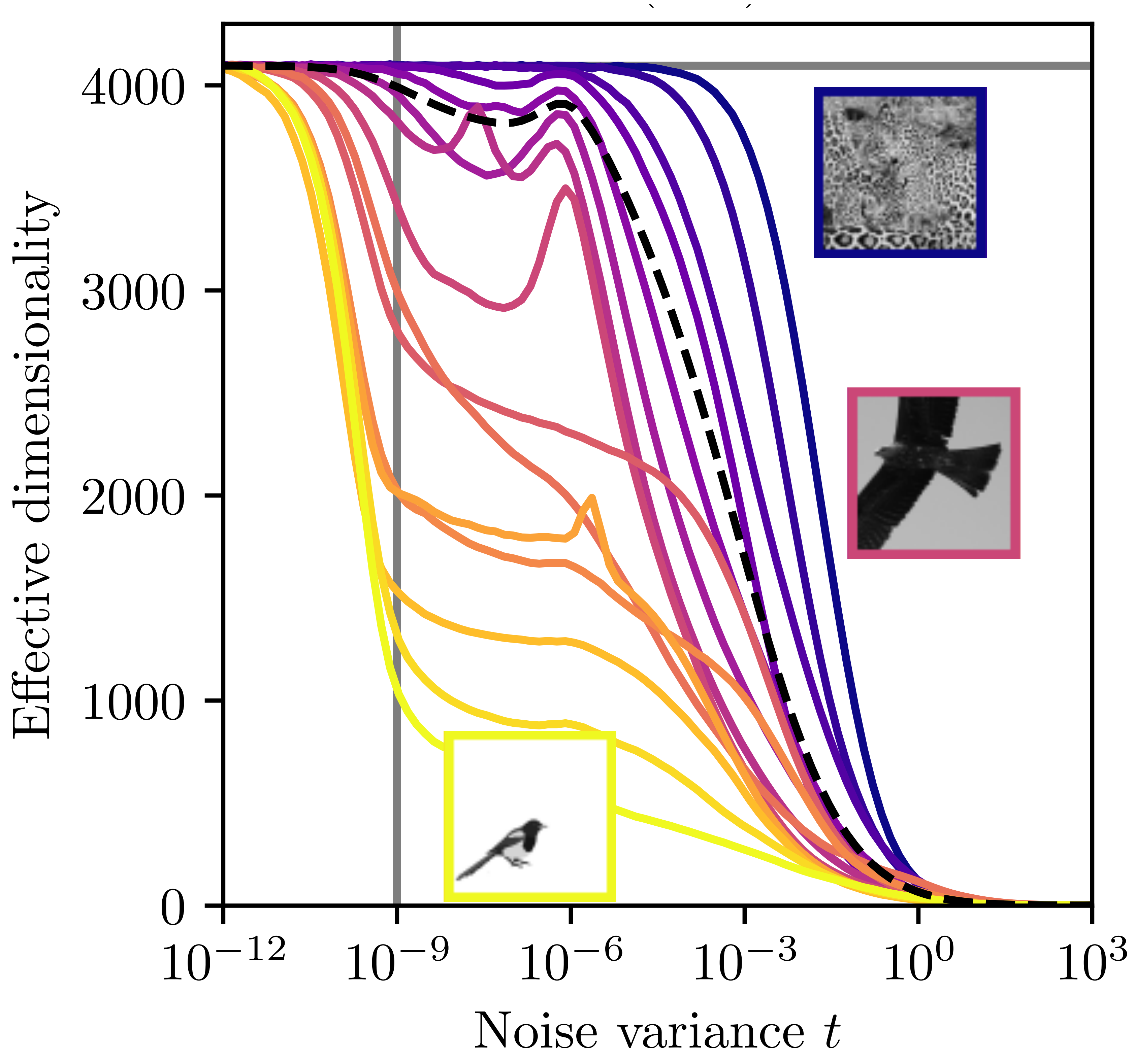


Testing the manifold hypothesis

Dimensionality is a scale-dependent-measure!

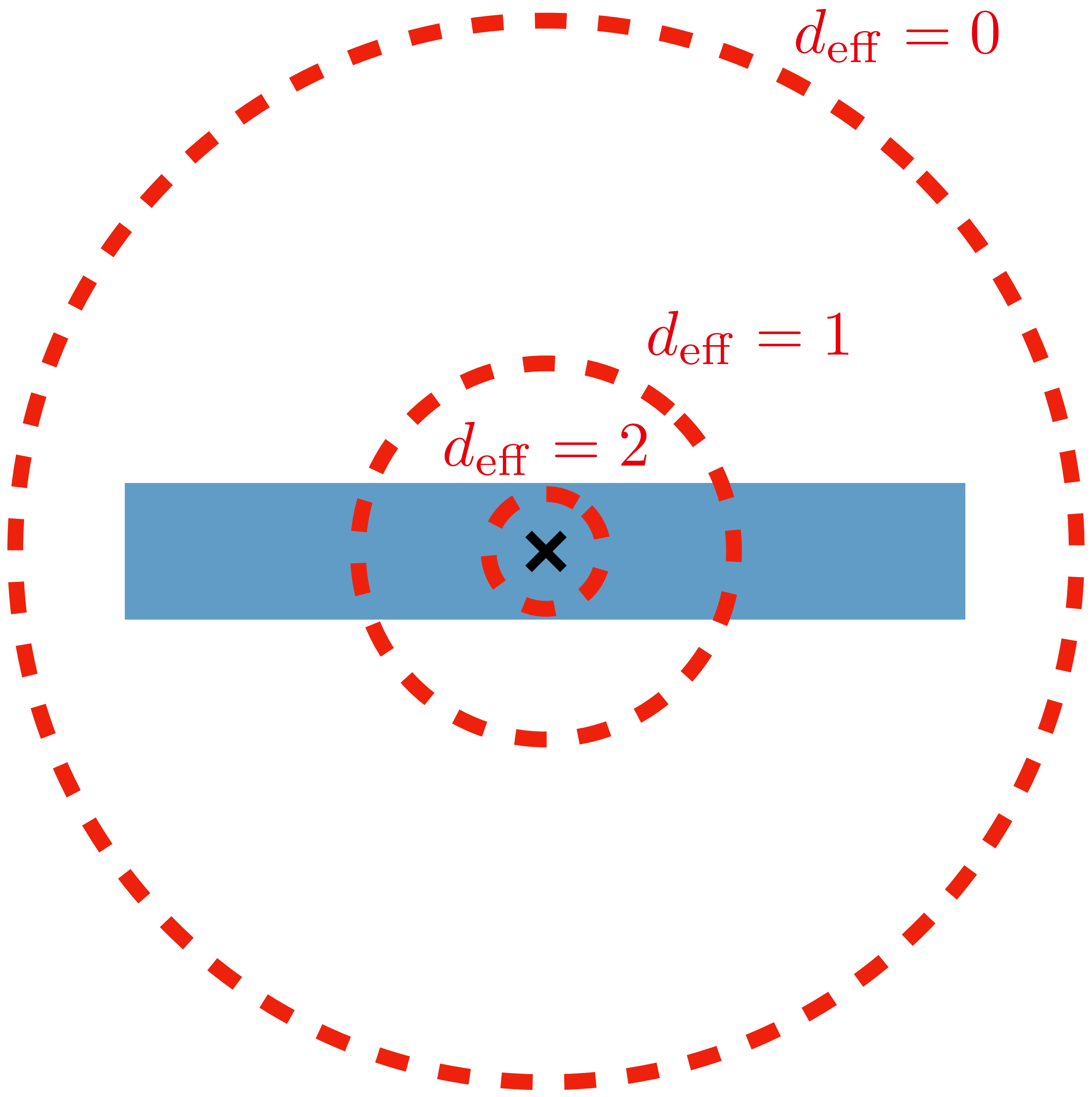


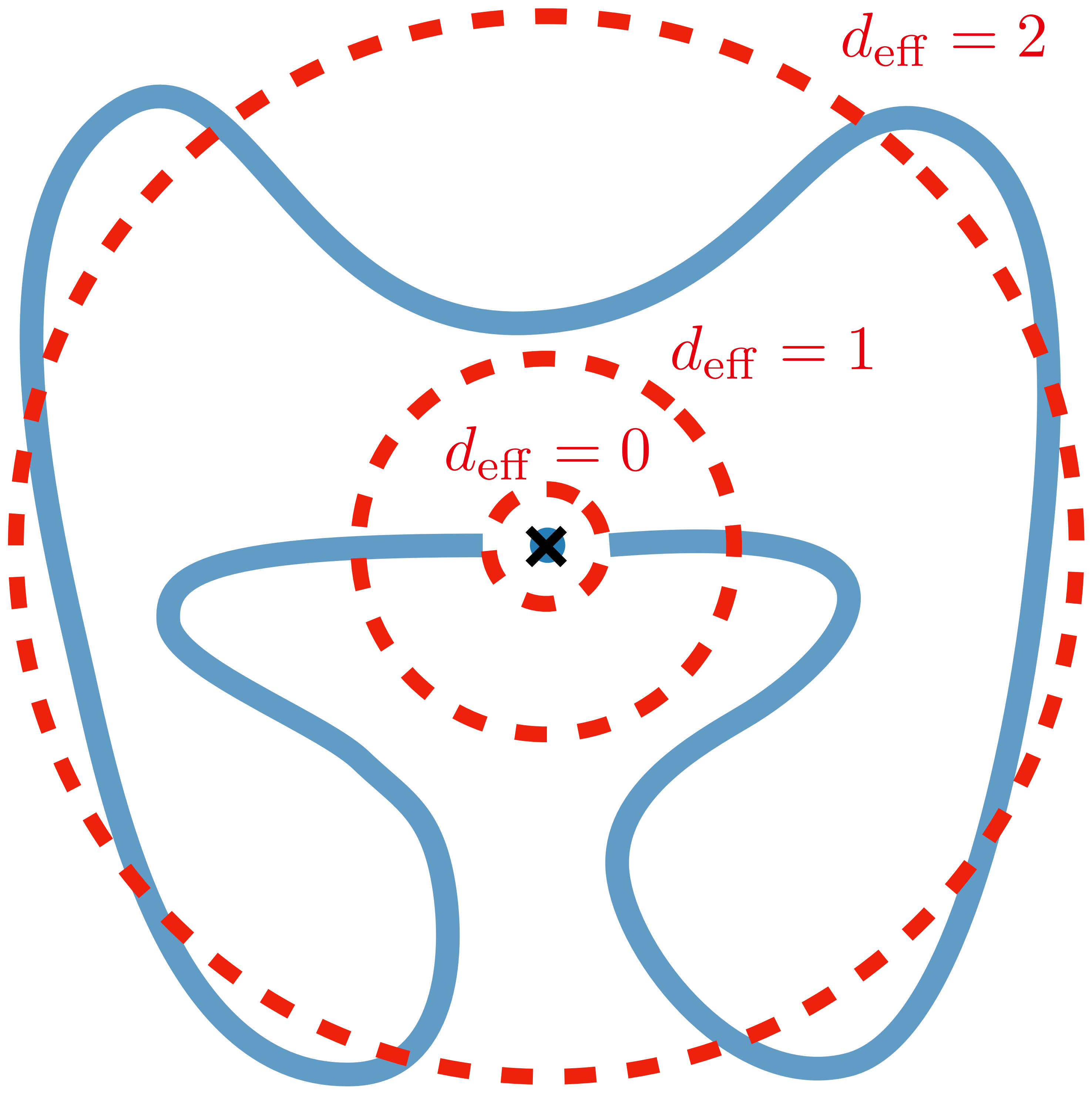


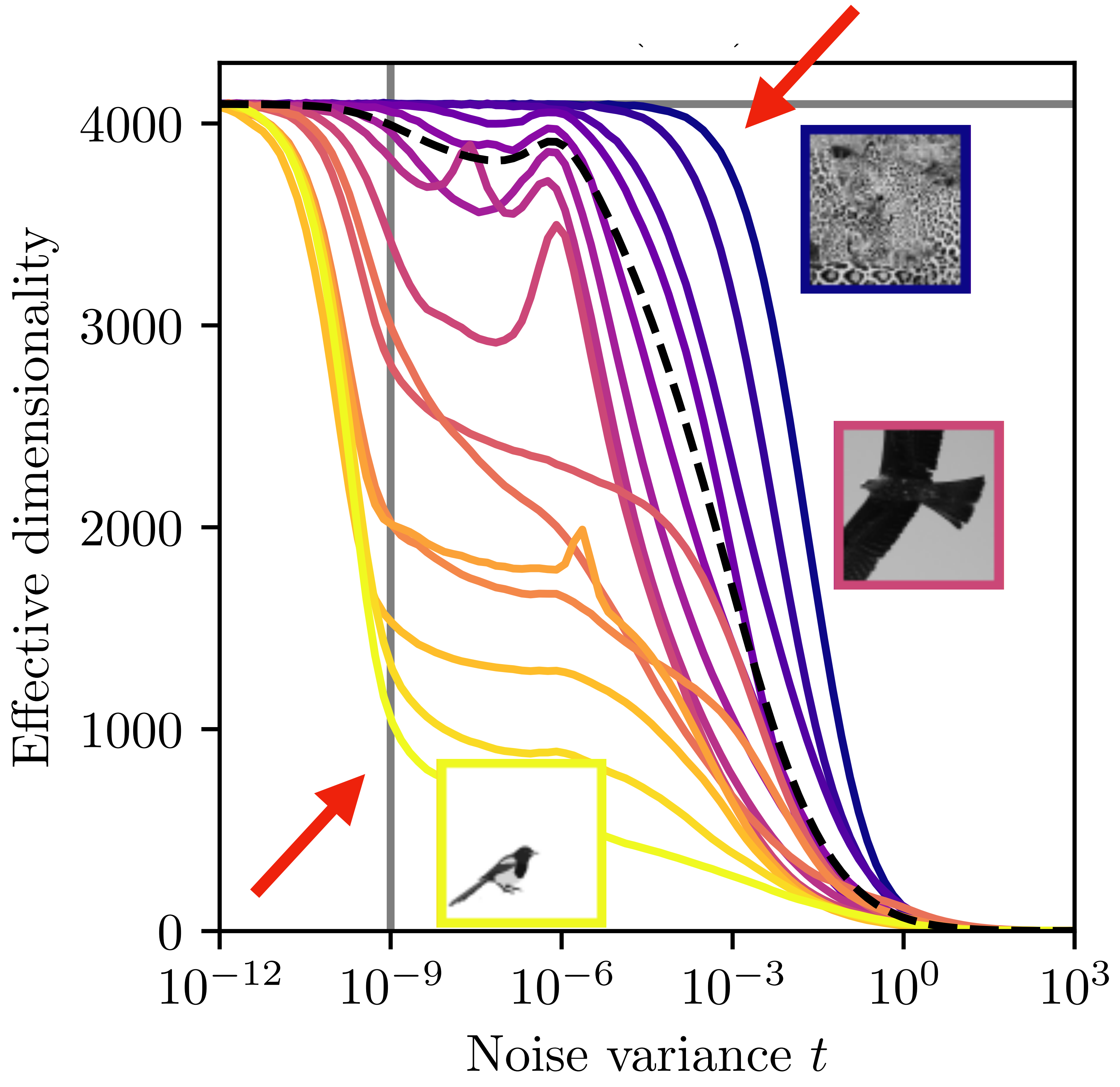


Testing the manifold hypothesis

Dimensionality is a scale-dependent-measure!







Summary

- We have access to a normalized density model of images!
- We can explore its **geometry**
- Surprising phenomena: lack of concentration, varying dimensionality
- **More properties of the landscape remain to be uncovered**

Summary

- We have access to a normalized density model of images!
- We can explore its **geometry**
- Surprising phenomena: lack of concentration, varying dimensionality
- **More properties of the landscape remain to be uncovered**

Thank you!