Data Free Training of Diffusion Models for Lattice Simulations

Shinichi Nakajima, Dennis Grinwald, Fadli Damara, Ankur Singha
Khaled Kahouli, Jakob Kauffmann, Marco Morik
BIFOLD, TU Berlin
Stefan Kühn, Karl Jansen
DESY Zeuthen

Recent projects

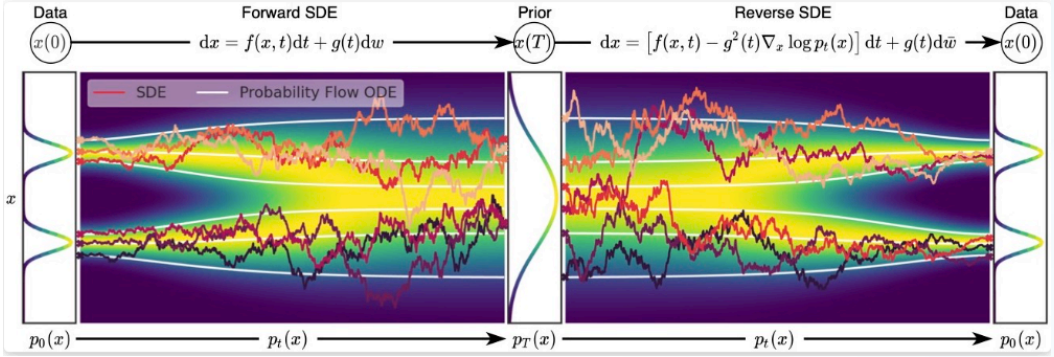
- Data free training of diffusion models for lattice field theories
  (Shinichi Nakajima, Dennis Grinwald: **this talk**)
- Conditional discrete diffusion for efficient disorder averaging in spin glasses
  (Fadli Damara: **this talk**)
- Multi-level sampling and topological-sector-resolved sampling for gauge theories
  (Ankur Singha: **next talk**)

Recent projects

- Data free training of diffusion models for lattice field theories
  (Shinichi Nakajima, Dennis Grinwald: **this talk**)
- Conditional discrete diffusion for efficient disorder averaging in spin glasses
  (Fadli Damara: **this talk**)
- Multi-level sampling and topological-sector-resolved sampling for gauge theories
  (Ankur Singha: **next talk**)

Diffusion Models

[Sohl-Dickstein+:arXiv:1503.03585, Ho+:arXiv:2006.11239, Song+:arXiv:2011.13456]
[Wang+:arXiv: 2309.17082, Vega+:arXiv:2510.26081, Aarts+:arXiv2601.19552,Tan+:arXiv:2607.08505]



- State-of-the-art in many domains (image, video, molecule, etc.)
- Fixed (forward) process -> Efficient training with score matching
- Exact sampling density of equivalent probability ODE
- Require training data

Data free training of lattice field theories?

Data free training methods

When action $S[\phi] = -\log p(\phi) + \text{const.}$ is accessible, diffusion models can be trained by

Annealing [Tomiya:arXiv:2607.12587, Schopmans+:arXiv:2501.19077]
generates training data from higher temperature models with MH-rejection or resampling

Iterated Denoising Energy Matching (iDEM) [Akhound-Sadegh+:arXiv:2402.06121]
uses importance weighted target score matching

Adjoint sampling (AS) [Havens+:arXiv:2504.11713]
uses target score matching (self-consistent fixed-point)

Score matching estimators (when training data are given)

Denoising Score Matching (DSM)

Diffusion model is trained by learning the score of Gaussian convoluted data distribution:

$$p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1)$$

where $p_0(\boldsymbol{x}) = \frac{e^{-S(\boldsymbol{x})}}{Z}$

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}_D(\boldsymbol{x}_t; \boldsymbol{x}_0, \sigma_t^2)$$

Denoising Score Matching (DSM)

Diffusion model is trained by learning the score of Gaussian convoluted data distribution:

$$p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1)$$

where $p_0(\boldsymbol{x}) = \frac{e^{-S(\boldsymbol{x})}}{Z}$

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}_D(\boldsymbol{x}_t; \boldsymbol{x}_0, \sigma_t^2)$$

(1) gives

$$\begin{aligned}
\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) &= \frac{1}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} p_t(\boldsymbol{x}_t) \\
&= \frac{1}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \\
&= \int \frac{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0 \\
&= \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\right],
\end{aligned}$$

Denoising Score Identity (DSI)
(standard score estimator)

Denoising Score Matching (DSM)

Diffusion model is trained by learning the score of Gaussian convoluted data distribution:

$$p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1) \qquad \text{where} \qquad p_0(\boldsymbol{x}) = \frac{e^{-S(\boldsymbol{x})}}{Z}$$

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}_D(\boldsymbol{x}_t; \boldsymbol{x}_0, \sigma_t^2)$$

(1) gives

$$\begin{aligned}
\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) &= \frac{1}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} p_t(\boldsymbol{x}_t) \\
&= \frac{1}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \\
&= \int \frac{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0 \\
&= \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) \right],
\end{aligned}$$

Denoising Score Identity (DSI)
(standard score estimator)

DSM minimizes

$$L^{\mathrm{DSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{Uniform}(t)} \left[\| \boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) \|^2 \right].$$

Target Score Matching (TSM)

[arXiv:2402.08667]

Diffusion model is trained by learning the score of Gaussian convoluted data distribution:

$$p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1) \qquad \text{where} \qquad p_0(\boldsymbol{x}) = \frac{e^{-S(\boldsymbol{x})}}{Z}$$

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}_D(\boldsymbol{x}_t; \boldsymbol{x}_0, \sigma_t^2)$$

Target Score Matching (TSM)

[arXiv:2402.08667]

Diffusion model is trained by learning the score of Gaussian convoluted data distribution:

$$p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1)$$

$$= \int \mathcal{N}_D(\boldsymbol{x}_t; \boldsymbol{x}_0, \sigma_t^2)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1')$$

$$= \int \mathcal{N}_D(\boldsymbol{\varepsilon}_t; \mathbf{0}, \sigma_t^2)p_0(\boldsymbol{x}_t - \boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t, \qquad (2)$$

where $p_0(\boldsymbol{x}) = \dfrac{e^{-S(\boldsymbol{x})}}{Z}$

$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}_D(\boldsymbol{x}_t; \boldsymbol{x}_0, \sigma_t^2)$

Target Score Matching (TSM)

[arXiv:2402.08667]

Diffusion model is trained by learning the score of Gaussian convoluted data distribution:

$$p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1)$$

$$= \int \mathcal{N}_D(\boldsymbol{x}_t;\boldsymbol{x}_0,\sigma_t^2)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \qquad (1')$$

$$= \int \mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t, \qquad (2)$$

where $p_0(\boldsymbol{x}) = \dfrac{e^{-S(\boldsymbol{x})}}{Z}$

$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}_D(\boldsymbol{x}_t;\boldsymbol{x}_0,\sigma_t^2)$

(2) gives

$$\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t) = \frac{1}{p_t(\boldsymbol{x}_t)}\int \mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)\nabla_{\boldsymbol{x}_t}p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t$$

$$= \frac{1}{p_t(\boldsymbol{x}_t)}\int \mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)\nabla_{\boldsymbol{x}_t}\log p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t$$

$$= \int \frac{\mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)}{p_t(\boldsymbol{x}_t)}\nabla_{\boldsymbol{x}_t}\log p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t$$

$$= \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}\left[\nabla_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\right]$$

Target Score Identity (TSI)
(Score estimated from action)

Target Score Matching (TSM)

[arXiv:2402.08667]

Diffusion model is trained by learning the score of Gaussian convoluted data distribution:

$$p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \quad (1)$$

$$= \int \mathcal{N}_D(\boldsymbol{x}_t;\boldsymbol{x}_0,\sigma_t^2)p_0(\boldsymbol{x}_0)d\boldsymbol{x}_0 \quad (1')$$

$$= \int \mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t, \quad (2)$$

where $p_0(\boldsymbol{x}) = \frac{e^{-S(\boldsymbol{x})}}{Z}$

$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}_D(\boldsymbol{x}_t;\boldsymbol{x}_0,\sigma_t^2)$

(2) gives

$$\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t) = \frac{1}{p_t(\boldsymbol{x}_t)}\int \mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)\nabla_{\boldsymbol{x}_t}p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t$$

$$= \frac{1}{p_t(\boldsymbol{x}_t)}\int \mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)\nabla_{\boldsymbol{x}_t}\log p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t$$

$$= \int \frac{\mathcal{N}_D(\boldsymbol{\varepsilon}_t;\boldsymbol{0},\sigma_t^2)p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)}{p_t(\boldsymbol{x}_t)}\nabla_{\boldsymbol{x}_t}\log p_0(\boldsymbol{x}_t-\boldsymbol{\varepsilon}_t)d\boldsymbol{\varepsilon}_t$$

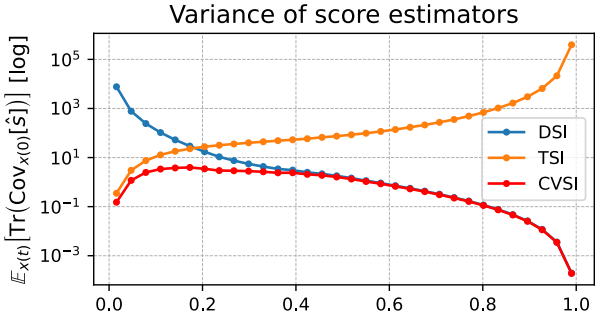
$$= \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}\left[\nabla_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\right]$$

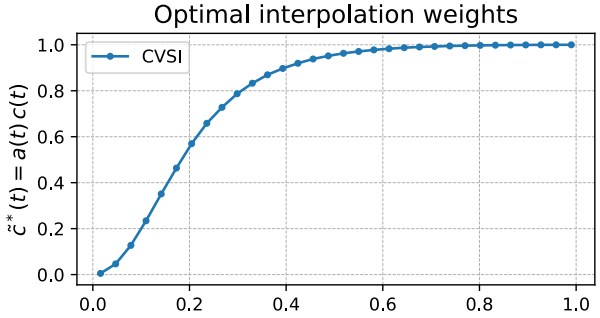
Target Score Identity (TSI)
(Score estimated from action)

TSM minimizes

$$L^{\mathrm{TSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\|^2\right].$$

Problems of DSM and TSM

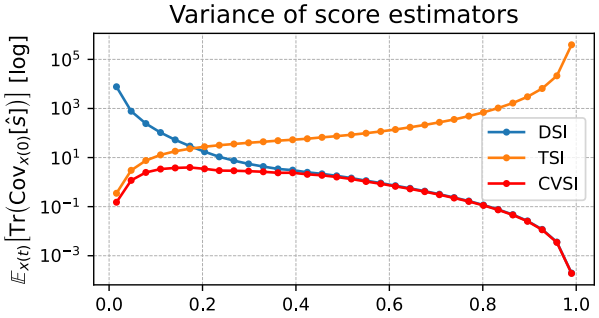


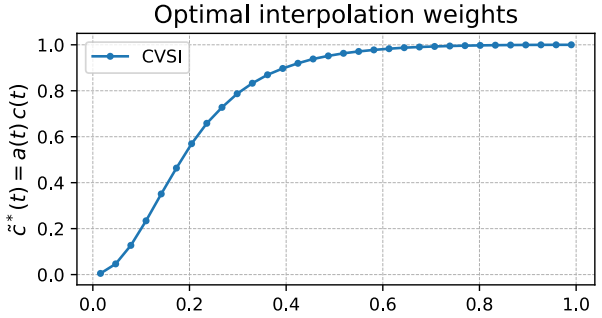


Variances of DSI and TSI estimator **diverges** at t=0 and t=1, respectively.

Control Variate Score Matching (CVSM)

[Kahouli+:arXiv:2512.20003]





Variances of DSI and TSI estimator **diverges** at t=0 and t=1, respectively.

CVSI mixes them with optimal balance:

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[c^*(t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) + (1 - c^*(t)) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \right].$$

with $c^*(t)$ analytically computed.

Relation to Force-regularization

[Vega et al., arXiv:2510.26081]

$$L^{\mathrm{DSM-FR}} = \underbrace{\mathbb{E}_{p(\boldsymbol{\phi}_t|\boldsymbol{\phi}_0)p(\boldsymbol{\phi}_0)\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{\phi}_t) - \boldsymbol{\nabla}_{\boldsymbol{\phi}_t} p(\boldsymbol{\phi}_t|\boldsymbol{\phi}_0)\|^2\right]}_{\text{DSM}} + \underbrace{\lambda_0 \mathbb{E}_{p(\boldsymbol{\phi}_0)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{\phi}_0) - \boldsymbol{\nabla}_{\boldsymbol{\phi}_0} \log p_0(\boldsymbol{\phi}_0)\|^2\right]}_{\text{Force regularizer}}.$$

Force regularization is a special case of mixing DSM and TSM, where TSM at t=0 is used.

Importance weighting for data-free training

Importance Weighting (IW)

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int \underbrace{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{DSI}$$

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int \underbrace{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{TSI}$$

Posterior distribution: $p(\boldsymbol{x}_0|\boldsymbol{x}_t) \propto p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)$

Draw $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$ then add noise $\boldsymbol{x}_t \sim p(\boldsymbol{x}_t|\boldsymbol{x}_0)$

Importance Weighting (IW)

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_0|\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{DSI}$$

$$= \int \underbrace{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2) \frac{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)}} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0$$

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_0|\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{TSI}$$

$$= \int \underbrace{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2) \frac{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)}} \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) d\boldsymbol{x}_0$$

Posterior distribution: $p(\boldsymbol{x}_0|\boldsymbol{x}_t) \propto p(\boldsymbol{x}_t|\boldsymbol{x}_0) p(\boldsymbol{x}_0)$

Draw $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$ then add noise $\boldsymbol{x}_t \sim p(\boldsymbol{x}_t|\boldsymbol{x}_0)$

Importance Weighting (IW)

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_0|\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{DSI}$$

$$= \int \mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2) \frac{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0$$

$$= \underbrace{\mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)}}_{\text{pink}} \underbrace{[w(\boldsymbol{x}_0; \boldsymbol{x}_t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0)]}_{\text{blue}},$$

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_0|\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{TSI}$$

$$= \int \mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2) \frac{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)} \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) d\boldsymbol{x}_0$$

$$= \underbrace{\mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)}}_{\text{pink}} \underbrace{[w(\boldsymbol{x}_0; \boldsymbol{x}_t) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0)]}_{\text{blue}},$$

where $\underbrace{w(\boldsymbol{x}_0; \boldsymbol{x}_t) = \frac{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)}}_{\text{Importance weight}}$

$\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$ not required!

Importance Weighting (IW)

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_0|\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{DSI}$$

$$= \int \mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2) \frac{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_0$$

$$= \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)} \left[w(\boldsymbol{x}_0; \boldsymbol{x}_t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) \right],$$

$$\approx \sum_{k=1}^{K} \underbrace{w^{(k)}} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0^{(k)}), \qquad \underbrace{\boldsymbol{x}_0^{(k)} \sim \mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)}$$

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_0|\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) d\boldsymbol{x}_0 \qquad : \text{TSI}$$

$$= \int \mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2) \frac{p(\boldsymbol{x}_0|\boldsymbol{x}_t)}{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)} \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) d\boldsymbol{x}_0$$

$$= \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)} \left[w(\boldsymbol{x}_0; \boldsymbol{x}_t) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \right],$$

$$\approx \sum_{k=1}^{K} \underbrace{w^{(k)}} \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0^{(k)}), \qquad \underbrace{\boldsymbol{x}_0^{(k)} \sim \mathcal{N}_D(\boldsymbol{x}_0; \boldsymbol{x}_t, \sigma_t^2)}$$

where $\underbrace{w^{(k)} = \frac{\exp(-U(\boldsymbol{x}_0^{(k)}))}{\sum_{k=1}^{K} \exp(-U(\boldsymbol{x}_0^{(k)}))}}$

Self-normalized Importance weight

$\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$ not required!

DSI and TSI hold even if $\boldsymbol{x}_t$ is not generated from $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$

DSM, TSM, IW-DSM, IW-TSM

$$L^{\mathrm{DSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right].$$

$$L^{\mathrm{TSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\|^2\right].$$

$$L^{\mathrm{IW-DSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)}\left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right],$$

$$L^{\mathrm{IW-TSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)}\left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\|^2\right].$$

IW estimator has too large variance!

DSM, TSM, IW-DSM, IW-TSM

Let $r_\theta(\boldsymbol{x}_0)$ be the Probability Flow ODE density with the current score estimate $\boldsymbol{s}_\theta(\boldsymbol{x}_t)$

$$L^{\mathrm{DSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)\underbrace{p(\boldsymbol{x}_0)}\mathrm{Uniform}(t)} \left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right].$$

$$L^{\mathrm{TSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)\underbrace{p(\boldsymbol{x}_0)}\mathrm{Uniform}(t)} \left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0)\|^2\right].$$

$$L^{\mathrm{IW-DSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)} \left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right],$$

$$L^{\mathrm{IW-TSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)} \left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0)\|^2\right].$$

IW estimator has too large variance!

Training with self-generated samples

Let $r_\theta(\boldsymbol{x}_0)$ be the Probability Flow ODE density with the current score estimate $\boldsymbol{s}_\theta(\boldsymbol{x}_t)$

Then, training objectives with the self generated data are

$$L^{\mathrm{DSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)\underbrace{r_\theta(\boldsymbol{x}_0)}\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right].$$

$$L^{\mathrm{TSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)\underbrace{r_\theta(\boldsymbol{x}_0)}\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\|^2\right].$$

$$L^{\mathrm{IW-DSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)}\left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right],$$

$$L^{\mathrm{IW-TSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)}\left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \boldsymbol{\nabla}_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\|^2\right].$$

IW estimator has too large variance!

Training with self-generated samples

Let $r_\theta(\boldsymbol{x}_0)$ be the Probability Flow ODE density with the current score estimate $\boldsymbol{s}_\theta(\boldsymbol{x}_t)$

Then, training objectives with the self generated data are

If $r_\theta(\boldsymbol{x}_0) = p(\boldsymbol{x}_0)$

$$L^{\mathrm{DSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)r_\theta(\boldsymbol{x}_0)\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right]. \quad \text{unbiased}$$

$$L^{\mathrm{TSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)r_\theta(\boldsymbol{x}_0)\mathrm{Uniform}(t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\|^2\right]. \quad \text{unbiased}$$

$$L^{\mathrm{IW-DSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)}\left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t}\log p(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right], \quad \text{unbiased}$$

$$L^{\mathrm{IW-TSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2)\mathrm{Uniform}(t)}\left[w(\boldsymbol{w}_0;\boldsymbol{w}_t)\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_0}\log p_0(\boldsymbol{x}_0)\|^2\right]. \quad \text{unbiased}$$

IW estimator has too large variance!

Training with self-generated samples

Let $r_\theta(\boldsymbol{x}_0)$ be the Probability Flow ODE density with the current score estimate $\boldsymbol{s}_\theta(\boldsymbol{x}_t)$

Then, training objectives with the self generated data are

	If $r_\theta(\boldsymbol{x}_0) = p(\boldsymbol{x}_0)$	$r_\theta(\boldsymbol{x}_0) \neq p(\boldsymbol{x}_0)$
$L^{\mathrm{DSM}} = \mathbb{E}_{p(\boldsymbol{x}_t\|\boldsymbol{x}_0) r_\theta(\boldsymbol{x}_0) \mathrm{Uniform}(t)} \left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t\|\boldsymbol{x}_0)\|^2 \right].$	unbiased	no training
$L^{\mathrm{TSM}} = \mathbb{E}_{p(\boldsymbol{x}_t\|\boldsymbol{x}_0) r_\theta(\boldsymbol{x}_0) \mathrm{Uniform}(t)} \left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0)\|^2 \right].$	unbiased	fixed point
$L^{\mathrm{IW-DSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2) \mathrm{Uniform}(t)} \left[w(\boldsymbol{w}_0;\boldsymbol{w}_t) \|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t\|\boldsymbol{x}_0)\|^2 \right],$	unbiased	unbiased
$L^{\mathrm{IW-TSM}} = \mathbb{E}_{\mathcal{N}_D(\boldsymbol{x}_0;\boldsymbol{x}_t,\sigma_t^2) \mathrm{Uniform}(t)} \left[w(\boldsymbol{w}_0;\boldsymbol{w}_t) \|\boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0)\|^2 \right].$	unbiased	unbiased

IW estimator has too large variance!

Data free training methods

When action $S[\phi] = -\log p(\phi) + \text{const.}$ is accessible, diffusion models can be trained by

Annealing [Tomiya:arXiv:2607.12587, Schopmans+:arXiv:2501.19077]
generates training data from higher temperature models with MH-rejection or resampling

Iterated Denoising Energy Matching (iDEM) [Akhound-Sadegh+:arXiv:2402.06121]
uses importance weighted target score matching

Adjoint sampling (AS) [Havens+:arXiv:2504.11713]
uses target score matching (self-consistent fixed-point)

Data free training methods

When action $S[\phi] = -\log p(\phi) + \text{const.}$ is accessible, diffusion models can be trained by

Annealing [Tomiya:arXiv:2607.12587, Schopmans+:arXiv:2501.19077]
generates training data from higher temperature models with MH-rejection or resampling

Iterated Denoising Energy Matching (iDEM) [Akhound-Sadegh+:arXiv:2402.06121]
uses importance weighted target score matching

Adjoint sampling (AS) [Havens+:arXiv:2504.11713]
uses target score matching (self-consistent fixed-point)

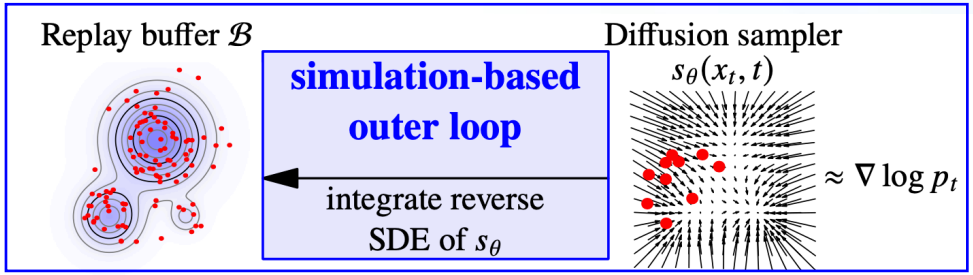
Our diffusion training with best choice of score estimator and noise schedule

- achieves comparable ESS to standard training of diffusion models on HMC samples.
- Its scalability will be compared with continuous normalizing flows (with Hutchinson density approx.).

Data-free training algorithm

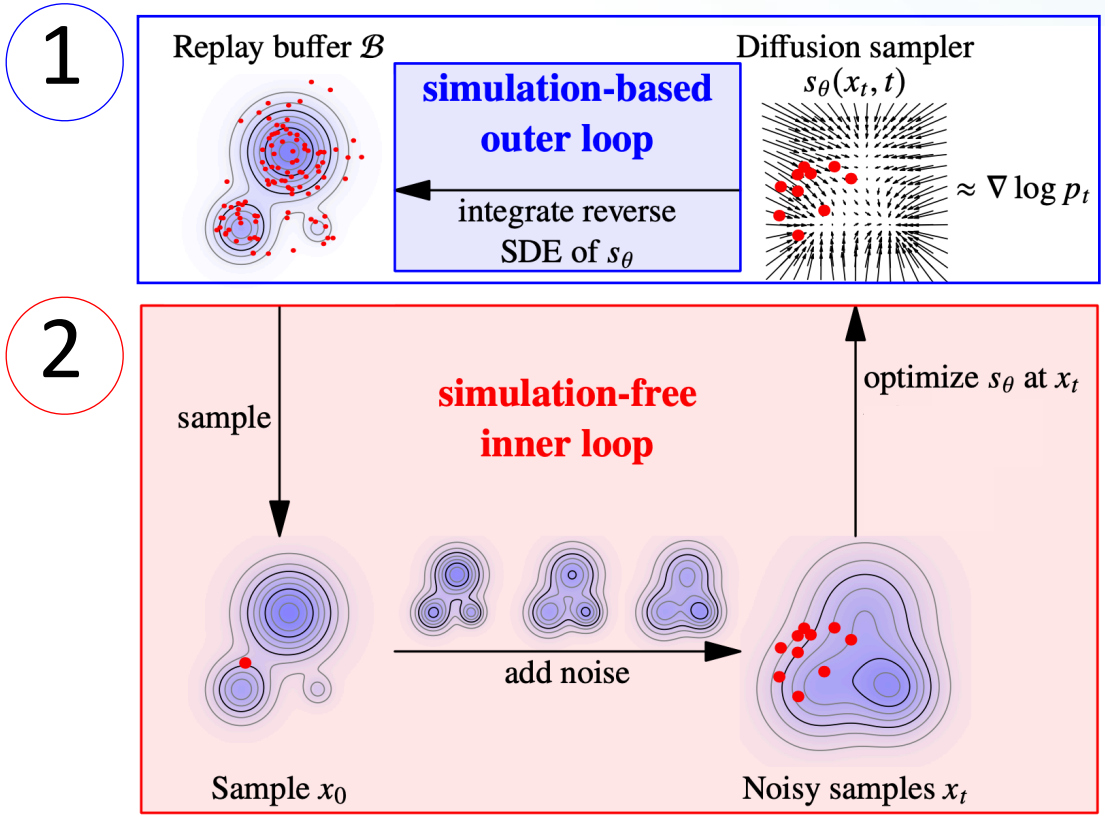
[Akhound-Sadegh+:arXiv:2402.06121]

1



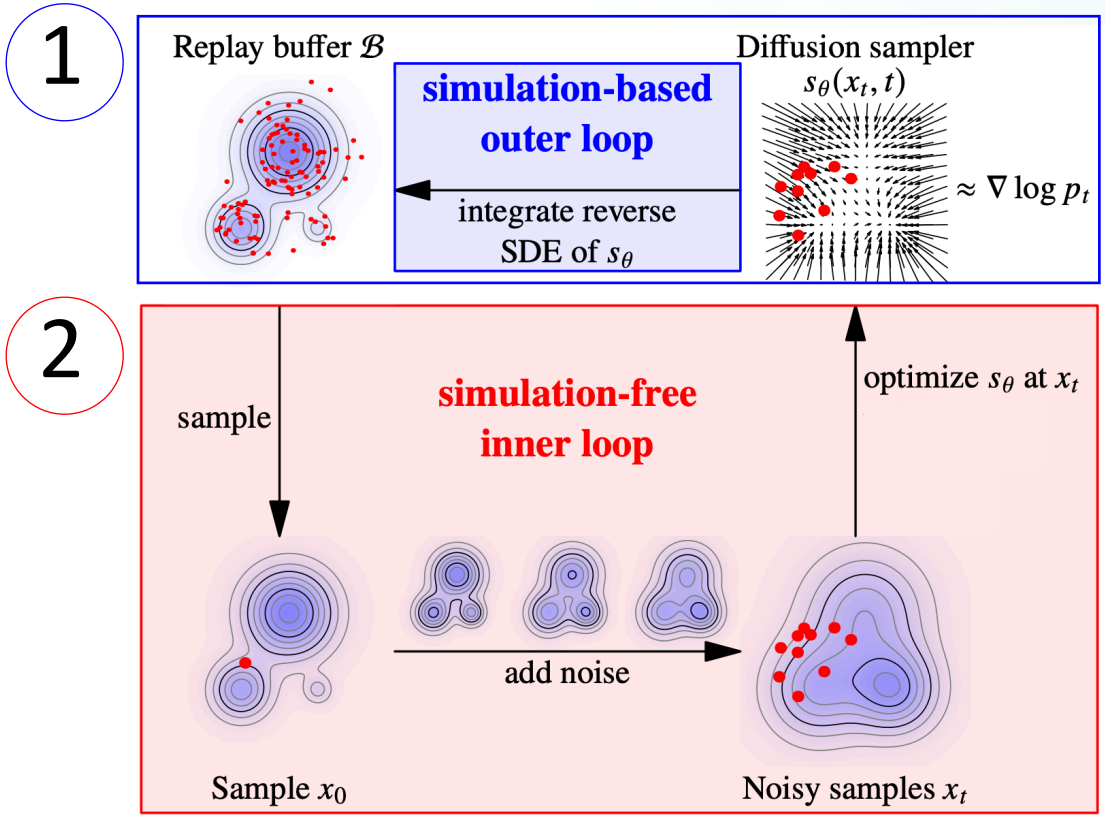
Data-free training algorithm

[Akhound-Sadegh+:arXiv:2402.06121]



Data-free training algorithm

[Akhound-Sadegh+:arXiv:2402.06121]



- Re-usage of buffered samples
- No trajectory simulation in the inner loop
- No gradient through an SDE solve

Noise schedule

[Song+:arXiv:2011.13456, Kahouli+:arXiv:2502.08598]

Perturbation kernel

$$p\big(\boldsymbol{\phi}(t) \,\big|\, \boldsymbol{\phi}(0)\big) = \mathcal{N}\Big(\boldsymbol{\phi}(t);\, a(t)\,\boldsymbol{\phi}(0),\; b^2(t)\,\mathbf{I}\Big)$$

Noise schedule

[Song+:arXiv:2011.13456, Kahouli+:arXiv:2502.08598]

Perturbation kernel

$$p\big(\boldsymbol{\phi}(t) \,\big|\, \boldsymbol{\phi}(0)\big) = \mathcal{N}\Big(\boldsymbol{\phi}(t);\, a(t)\,\boldsymbol{\phi}(0),\; b^2(t)\,\mathbf{I}\Big)$$

Sampling

$$\boldsymbol{\phi}(t) = a(t)\,\boldsymbol{\phi}(0) + b(t)\,\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Noise schedule

[Song+:arXiv:2011.13456, Kahouli+:arXiv:2502.08598]

Perturbation kernel

$$p\big(\boldsymbol{\phi}(t) \,\big|\, \boldsymbol{\phi}(0)\big) = \mathcal{N}\Big(\boldsymbol{\phi}(t);\, a(t)\,\boldsymbol{\phi}(0),\; b^2(t)\,\mathbf{I}\Big)$$

Sampling

$$\boldsymbol{\phi}(t) = a(t)\,\boldsymbol{\phi}(0) + b(t)\,\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Signal-to-noise ratio (SNR)

$$\gamma(t) = \frac{a(t)}{b(t)}$$

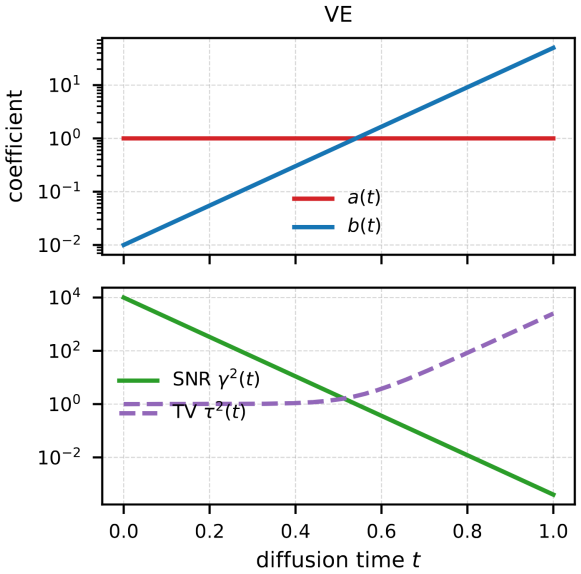
Total variance (TV)

$$\tau^2(t) = a^2(t) + b^2(t)$$

Noise schedule

[Song+:arXiv:2011.13456, Kahouli+:arXiv:2502.08598]

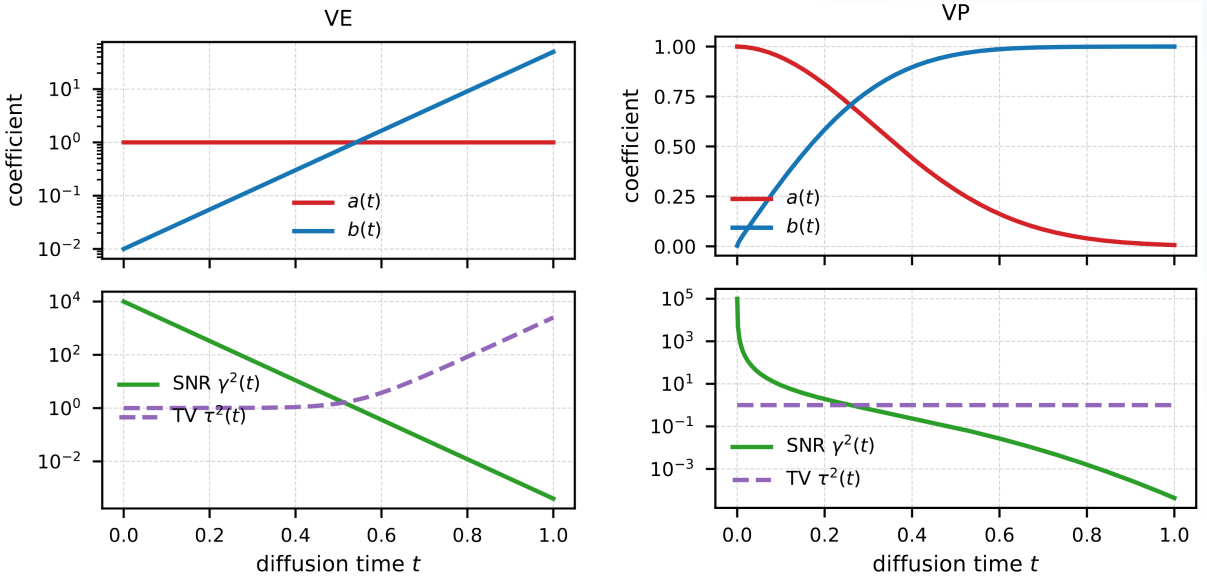
$$\boldsymbol{\phi}(t) = a(t)\,\boldsymbol{\phi}(0) + b(t)\,\boldsymbol{\epsilon} \qquad \text{SNR } \gamma(t) = a(t)/b(t) \qquad \text{TV } \tau^2(t) = a^2(t) + b^2(t)$$



Noise schedule

[Song+:arXiv:2011.13456, Kahouli+:arXiv:2502.08598]

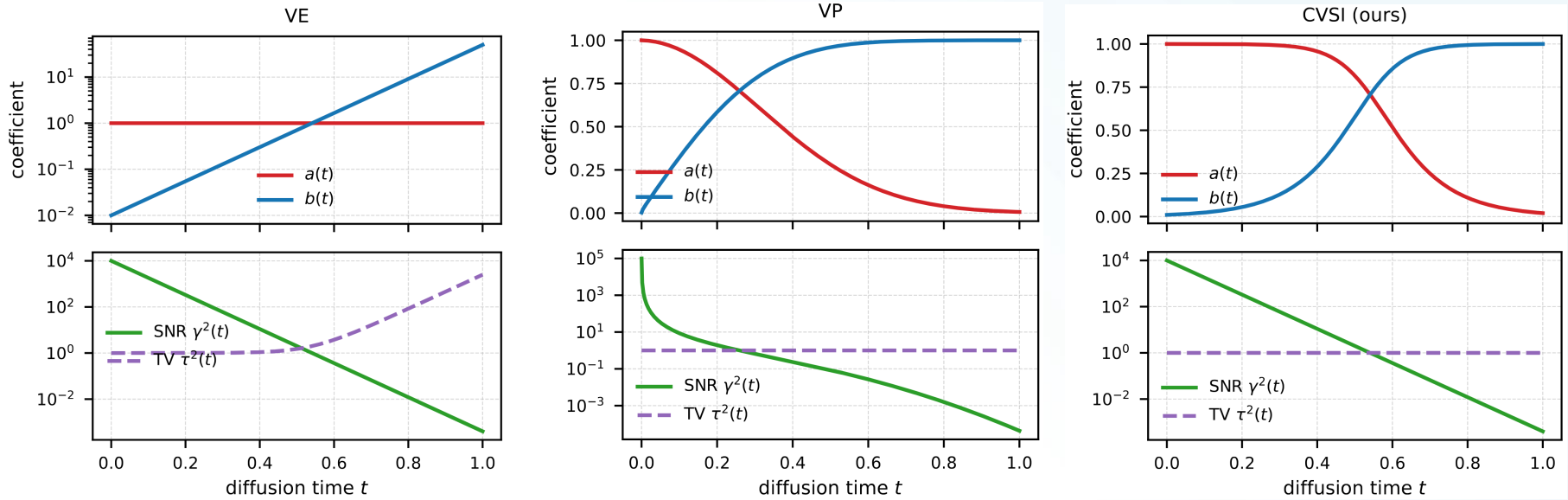
$$\boldsymbol{\phi}(t) = a(t)\,\boldsymbol{\phi}(0) + b(t)\,\boldsymbol{\epsilon} \qquad \text{SNR } \gamma(t) = a(t)/b(t) \qquad \text{TV } \tau^2(t) = a^2(t) + b^2(t)$$



Noise schedule

[Song+:arXiv:2011.13456, Kahouli+:arXiv:2502.08598]

$$\boldsymbol{\phi}(t) = a(t)\,\boldsymbol{\phi}(0) + b(t)\,\boldsymbol{\epsilon} \qquad \text{SNR } \gamma(t) = a(t)/b(t) \qquad \text{TV } \tau^2(t) = a^2(t) + b^2(t)$$



Loss target

[Kahouli+:arXiv:2512.20003]

Optimal target variance blend of two estimators, each clipped before blending

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[c^*(t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) + (1 - c^*(t)) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \right].$$

Loss target

[Kahouli+:arXiv:2512.20003]

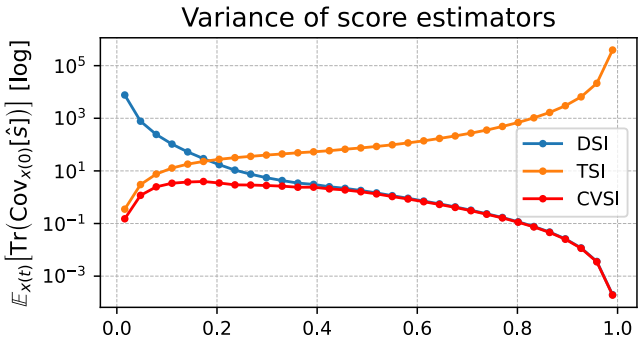
Optimal target variance blend of two estimators, each clipped before blending

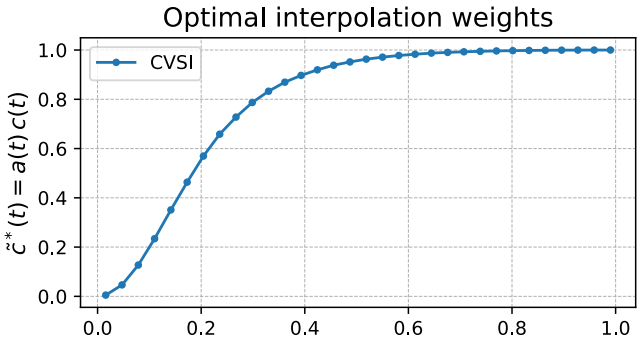
$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[c^*(t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) + (1 - c^*(t)) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \right].$$

$$c^*(t) = \frac{b^2(t)}{b^2(t) + a^2(t)\,\sigma_{\text{data}}^2} = \frac{1}{1 + \gamma^2(t)\,\sigma_{\text{data}}^2}$$

Loss target

[Kahouli+:arXiv:2512.20003]



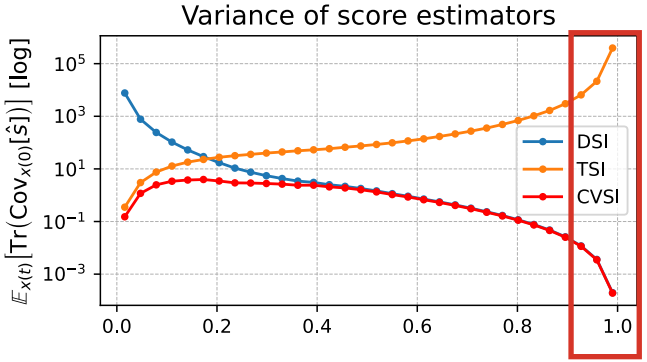


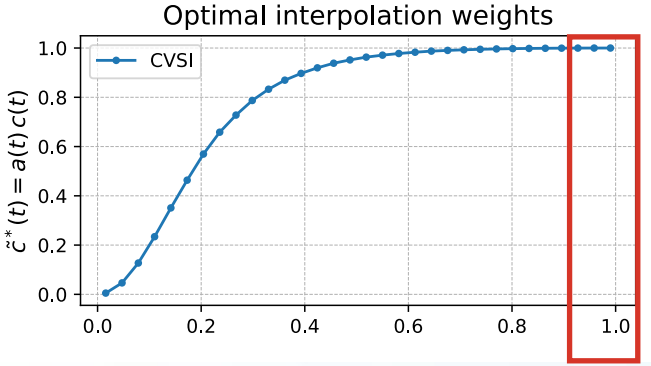
$$c^*(t) = \frac{b^2(t)}{b^2(t) + a^2(t)\,\sigma_{\text{data}}^2} = \frac{1}{1 + \gamma^2(t)\,\sigma_{\text{data}}^2}$$

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[c^*(t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) + (1 - c^*(t)) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \right].$$

Loss target

[Kahouli+:arXiv:2512.20003]





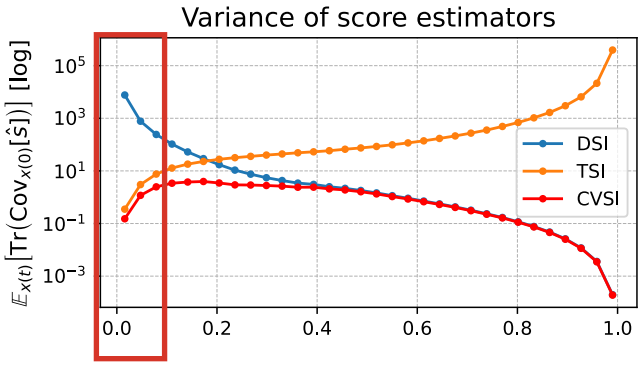
$$c^*(t) = \frac{b^2(t)}{b^2(t) + a^2(t)\,\sigma_{\text{data}}^2} = \frac{1}{1 + \gamma^2(t)\,\sigma_{\text{data}}^2}$$

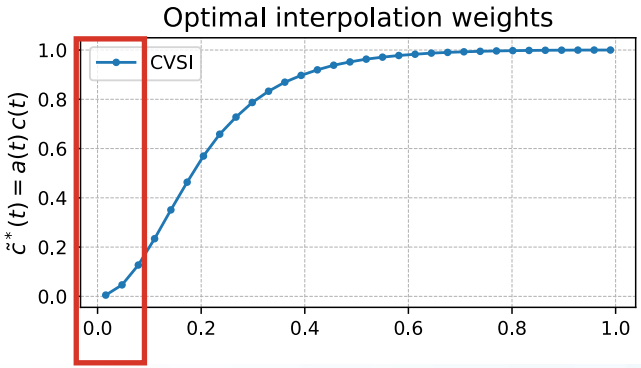
$$\boldsymbol{\nabla}_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[c^*(t) \boldsymbol{\nabla}_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) + (1 - c^*(t)) \boldsymbol{\nabla}_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \right].$$

As t → 1, a(t) → 0, b(t) → 1, c* → 1, target consists of mostly DSI term

Loss target

[Kahouli+:arXiv:2512.20003]





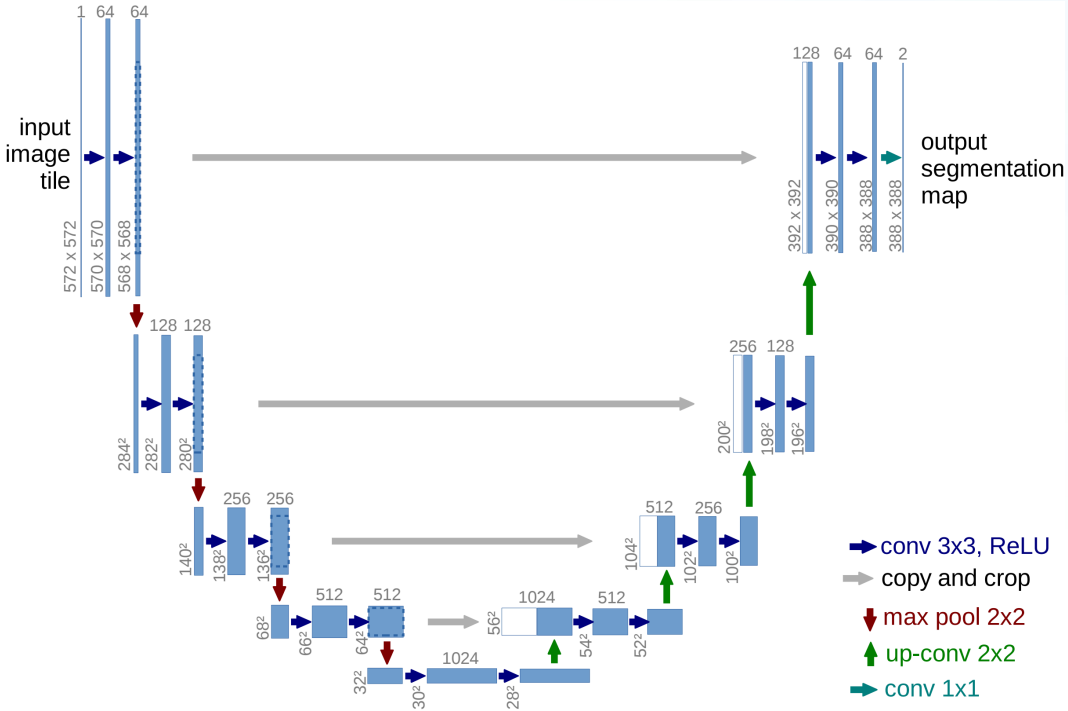
$$c^*(t) = \frac{b^2(t)}{b^2(t) + a^2(t)\,\sigma_{\text{data}}^2} = \frac{1}{1 + \gamma^2(t)\,\sigma_{\text{data}}^2}$$

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) = \mathbb{E}_{p(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[c^*(t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) + (1 - c^*(t)) \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \right].$$

As t → 0, a(t) → 1, b(t) → 0, c* → 0, target consists of mostly TSI term

Architecture

[Ronneberger+:arXiv:1505.04597, Vega+:arXiv:2510.26081]



Encoded symmetries

Translation: weight-shared convolutions

Periodicity: circular padding

$Z_2 : \phi \to -\phi$: anti-symmetrised score

$$\mathbf{s}_{\boldsymbol{\theta}}(\phi, t) \longleftarrow \frac{1}{2}\left(\mathbf{s}_{\boldsymbol{\theta}}(\phi, t) - \mathbf{s}_{\boldsymbol{\theta}}(-\phi, t)\right)$$

Experimental setup

Two-dimensional ϕ^4 theory

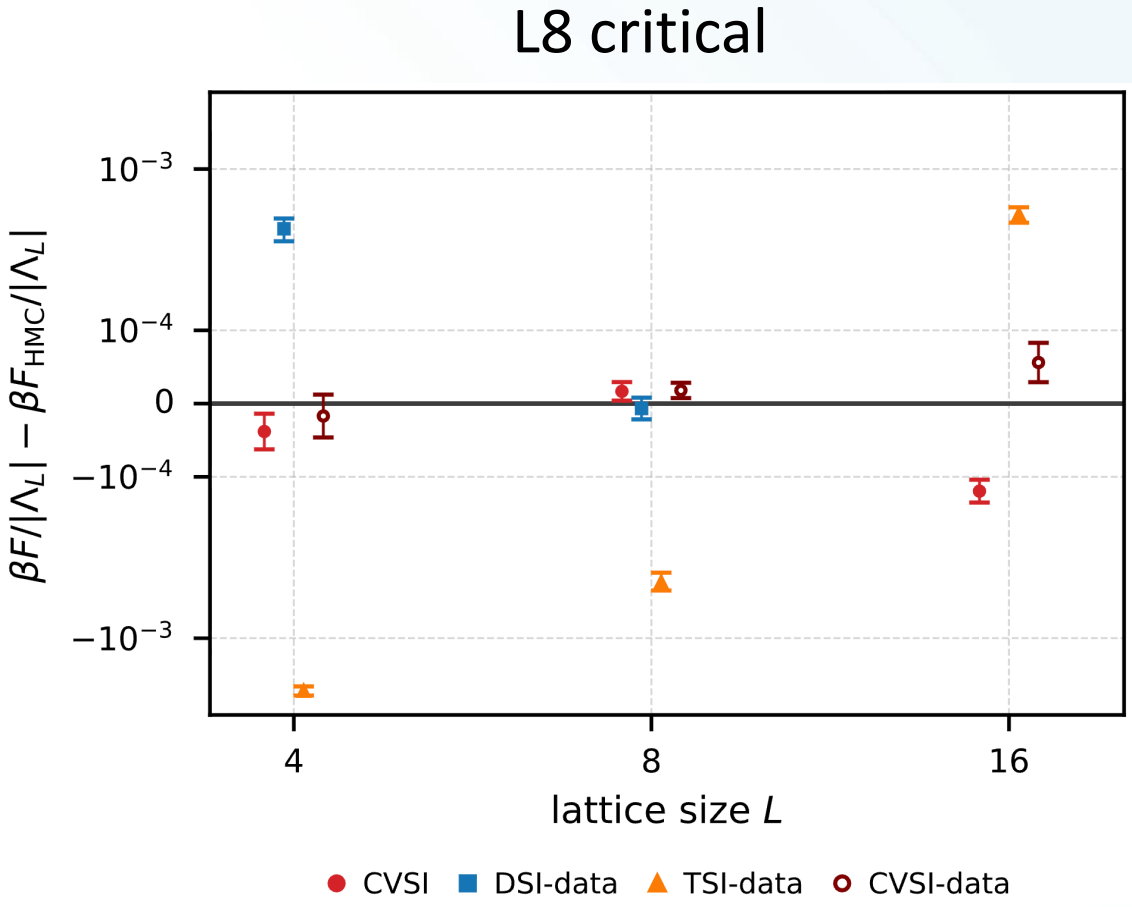
$$S[\boldsymbol{\phi}] = \sum_{\mathbf{x}} \Big[-2\kappa \sum_{\mu} \phi_{\mathbf{x}}\, \phi_{\mathbf{x}+\hat{\boldsymbol{\mu}}} + (1-2\lambda)\, \phi_{\mathbf{x}}^2 + \lambda\, \phi_{\mathbf{x}}^4 \Big] , \qquad p(\boldsymbol{\phi}) = \frac{1}{Z}\, e^{-S[\boldsymbol{\phi}]}$$

$$D = 2\,, \quad \Lambda_L = \{1, \ldots, L\}^2 \text{ periodic}\,, \quad L \in \{4, 8, 16\}$$

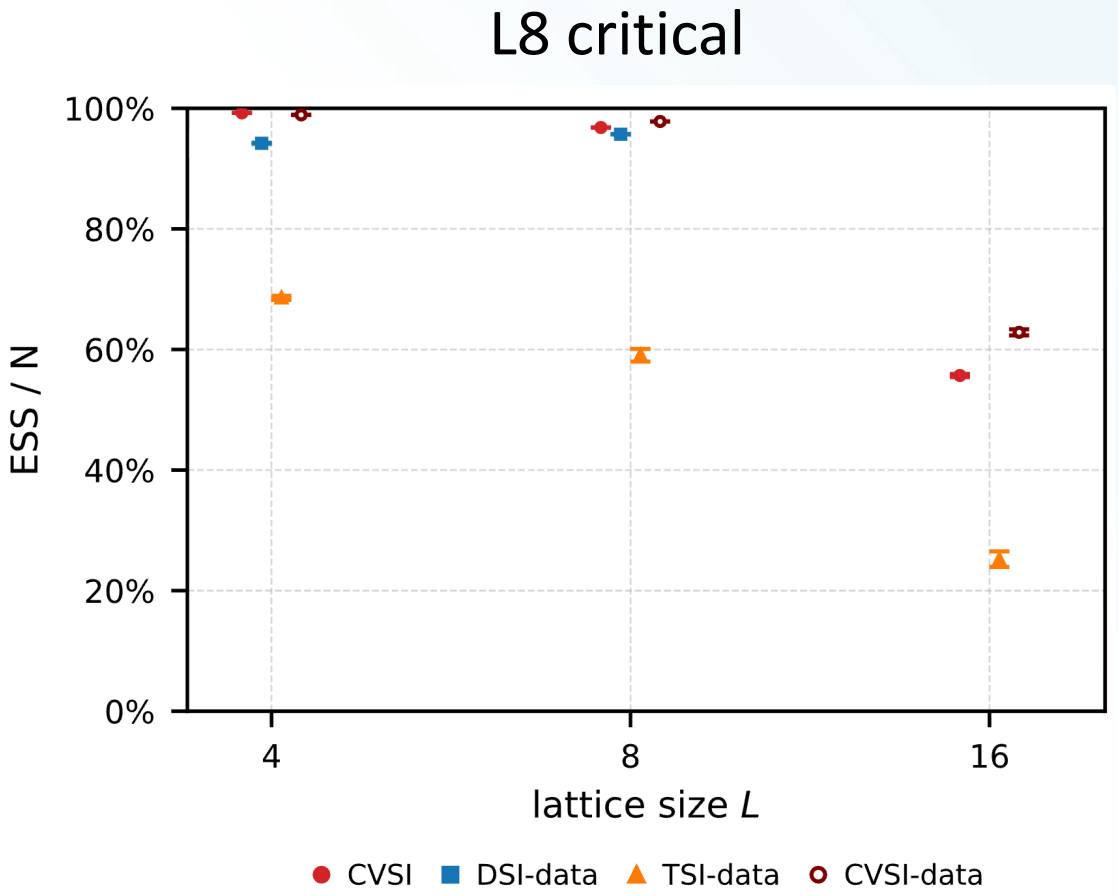
phase	κ	λ
symmetric	0.25	0.022
critical	0.2705	0.022
broken	0.28	0.022

HMC reference of 10^5 configurations each

Free energy comparison



ESS comparison

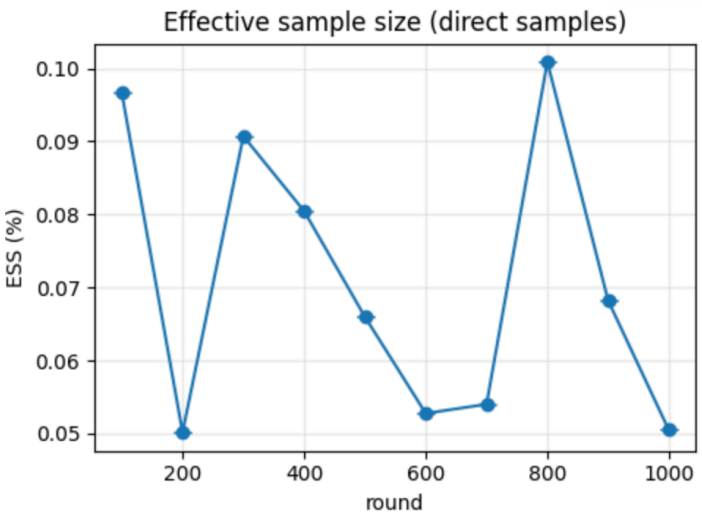


Results across phases & sizes

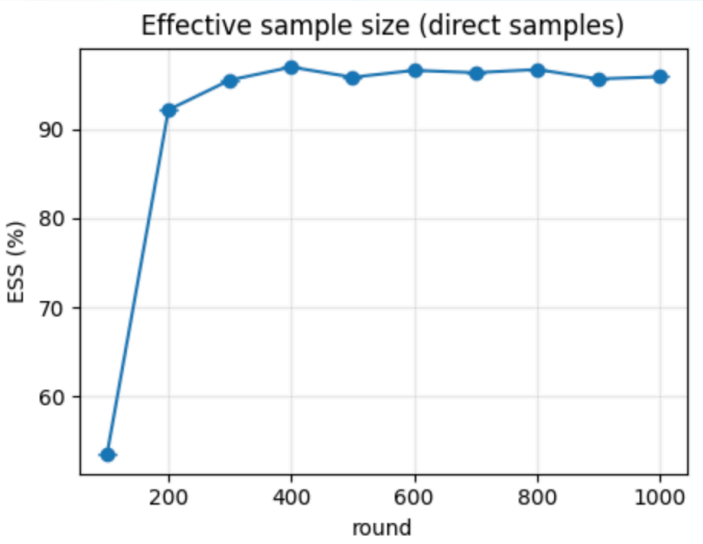
L	phase	$\beta F/\lvert\Lambda_L\rvert$	f_{TI} (HMC)	ESS (%)
4	symmetric	$-0.687155(13)$	$-0.687127(39)$	99.8
	critical	$-0.759648(24)$	$-0.759610(54)$	99.2
	broken	$-0.828094(24)$	$-0.827985(67)$	99.3
8	symmetric	$-0.670431(5)$	$-0.670380(17)$	99.5
	critical	$-0.721231(13)$	$-0.721248(25)$	96.8
	broken	$-0.787029(11)$	$-0.787052(35)$	97.4
16	symmetric	$-0.669316(18)$	$-0.669511(9)$	96.1
	critical	$-0.713179(18)$	$-0.713111(13)$	55.8
	broken	$-0.778655(120)$	$-0.778651(20)$	34.7

IW vs. No-IW ESS

IW CVSI



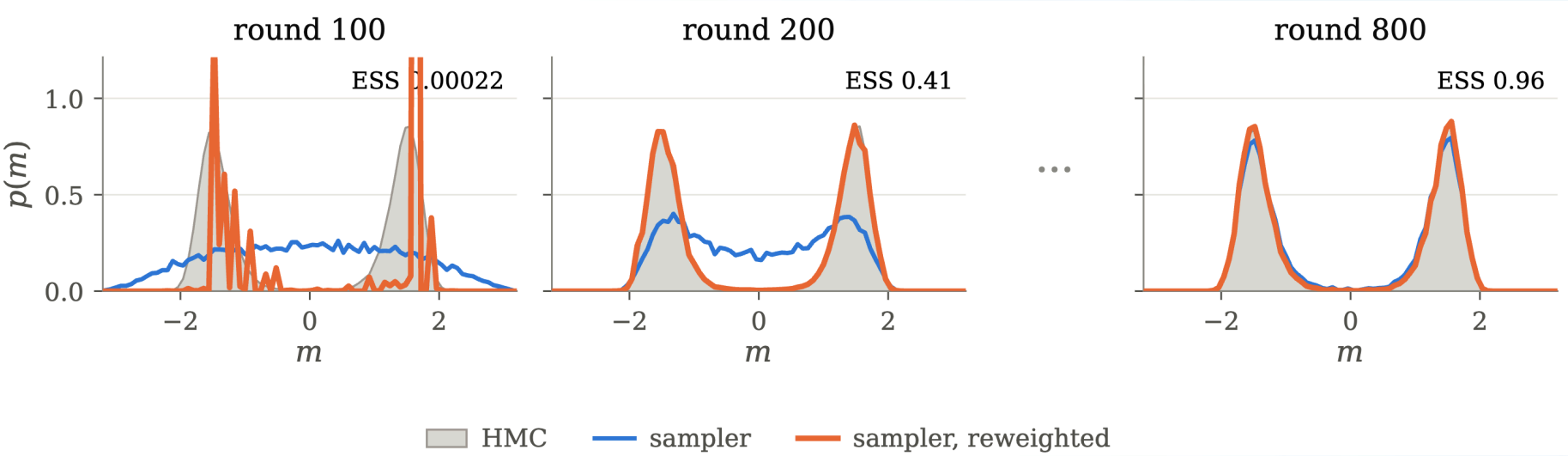
CVSI (ours)



L8 critical

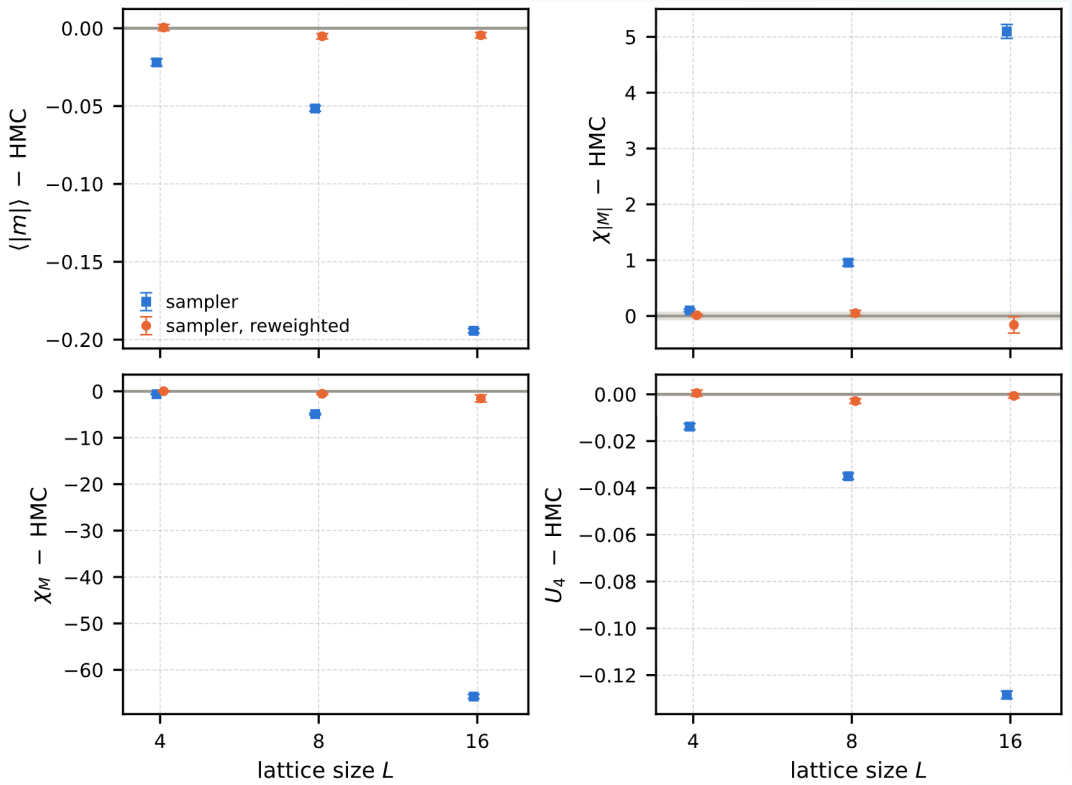
Magnetisation dist. throughout training

L8 broken

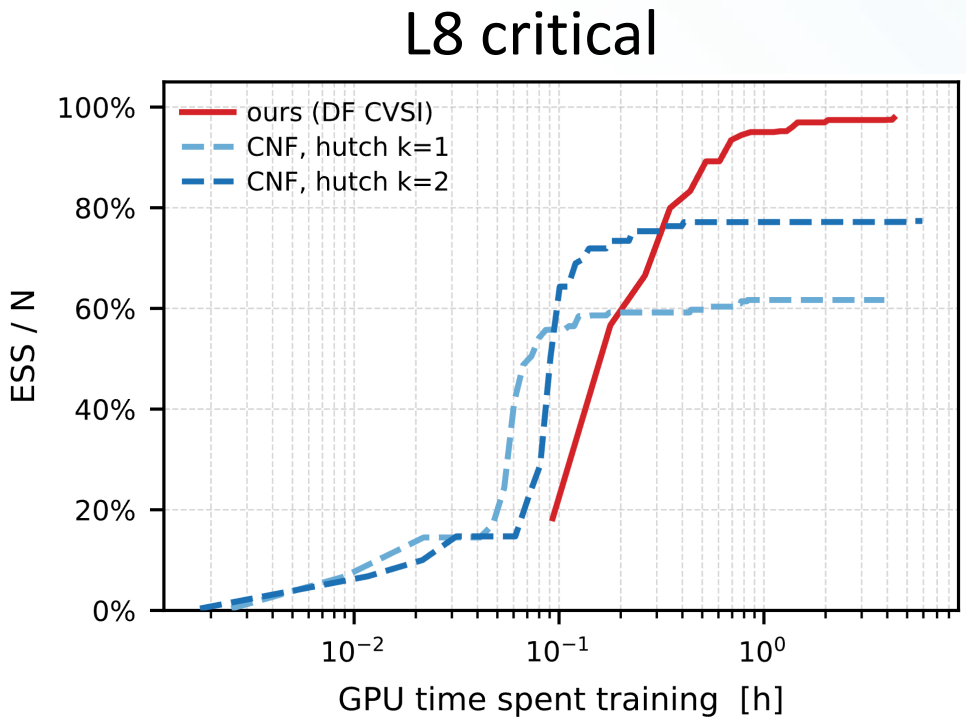


Other observables

L8 critical



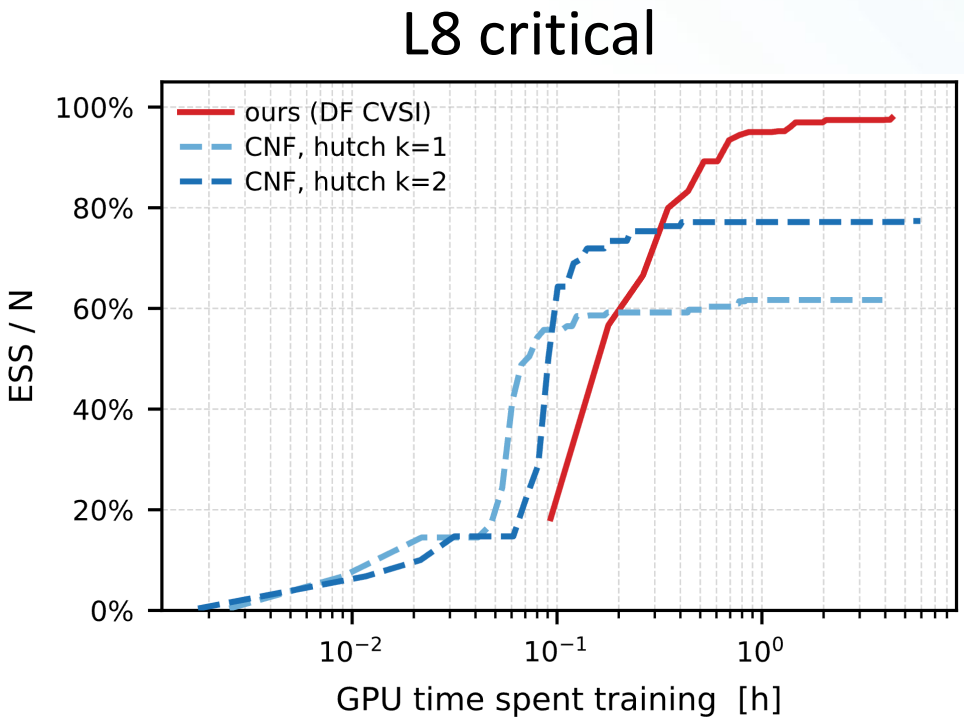
Comparison with CNF



- matched capacity: 7,005 (Conv-CNF) against 7,082 (UNet) parameters

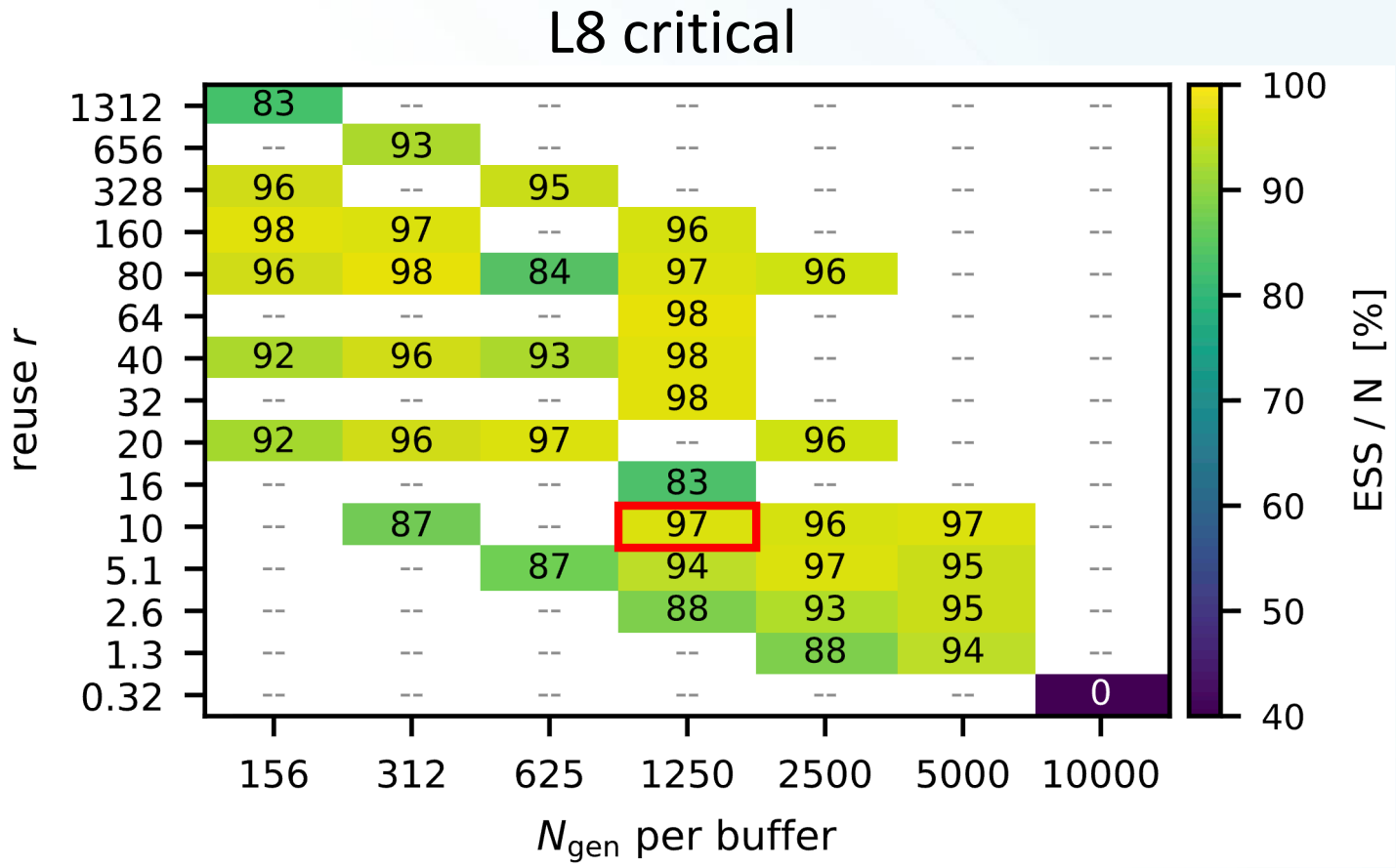
- CNF computes uses Hutchinson trace estimator(k = 1,2)

Comparison with CNF



- matched capacity: 7,005 (Conv-CNF) against 7,082 (UNet) parameters
- CNF computes uses Hutchinson trace estimator(k = 1,2)
- We achieve overall better ESS compared to CNF
- Whether we can converge faster across ESS dim. yet to be explored

Potential fix -> Tune reuse and generation



Conclusion

- We propose data-free training of a diffusion model with CVSI loss target and schedule, and w/o Importance Weighting on lattice Phi4 theory

Conclusion

- We propose data-free training of a diffusion model with CVSI loss target and schedule, and w/o Importance Weighting on lattice Phi4 theory

- Our approach achieves ESS comparable to training on MCMC data using DSI/CVSI target

Conclusion

- We propose data-free training of a diffusion model with CVSI loss target and schedule, and w/o Importance Weighting on lattice Phi4 theory
- Our approach achieves ESS comparable to training on MCMC data using DSI/CVSI target
- Importance weighting harms training
- CVIS contributes a lot to data-free training

Conclusion

- We propose data-free training of a diffusion model with CVSI loss target and schedule, and w/o Importance Weighting on lattice Phi4 theory
- Our approach achieves ESS comparable to training on MCMC data using DSI/CVSI target
- Importance weighting harms training
- CVIS contributes a lot to data-free training
- Computational benefit to CNF still evaluated completely, potential opt. axes are sample reusage and buffer size

Recent projects

- Data free training of diffusion models for lattice field theories
  [Grinwald+:arXiv:****.*****] (Shinichi Nakajima, Dennis Grinwald: this talk)
- Conditional discrete diffusion for efficient disorder averaging in spin glasses
  [Damara+:arXiv:****.*****] (Fadli Damara: this talk)
- Multi-level sampling and topological-sector-resolved sampling for gauge theories
  [Singha+:arXiv****.*****] (Ankur Singha: today at 15:45)

Conditional Discrete Diffusion for Efficient Disorder Averaging in Spin Glasses

Spin Glass Model:

$$H_J(\sigma) = -\sum_{\langle ij \rangle} J_{ij}\sigma_i\sigma_j \qquad J_{ij} \sim \mathcal{N}(0,1)$$

In our work we consider:

Edwards-Anderson spin glass (nearest neighbors couplings)

Disorder Averaged Observable:

$$\left[\langle O \rangle\right] = \mathbb{E}_J\left[\langle O \rangle\right]$$

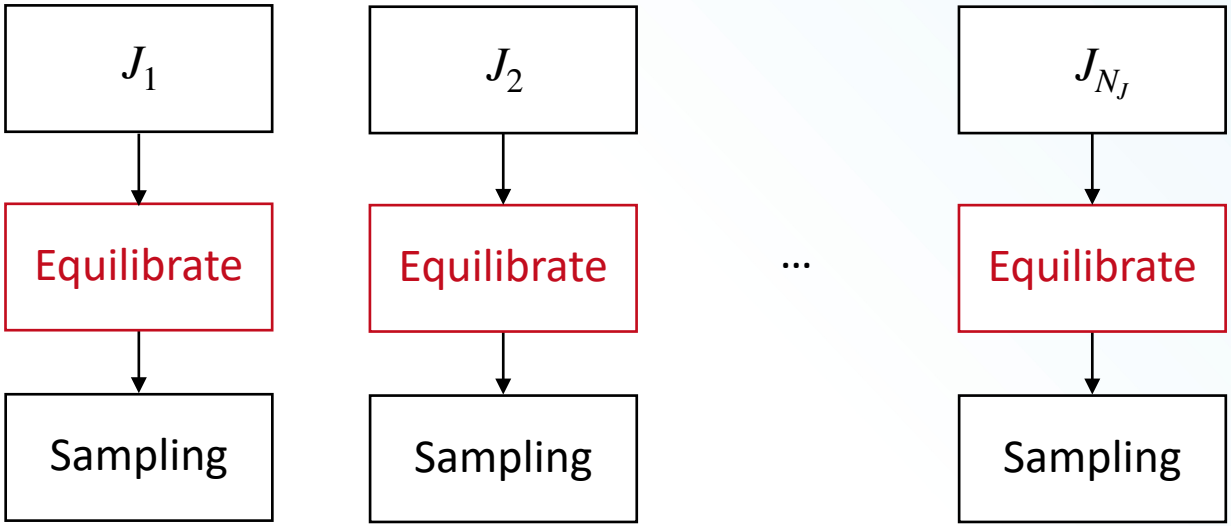
Examples of interesting observables:

Overlap q_{EA} , Reentrance behavior Ψ (PM -> FM -> PM)

Generating training data for every J is expensive!

MCMC requires equilibration for every J

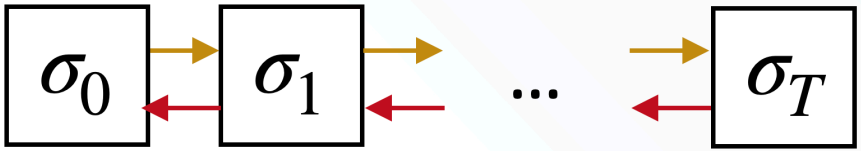
For **one** instantiation of disorder J:



Data Free Training
Of Discrete Diffusion Models

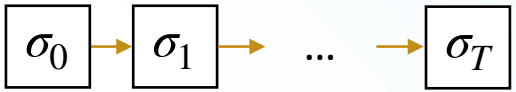
Data-Free Training: Part 1 (Discrete Diffusion)

Forward corruption process (fixed)



Reverse denoising process (learned)

Data-Free Training: Part 1 (Discrete Diffusion)



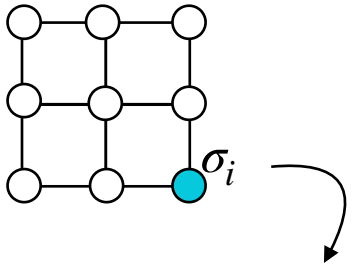
Forward process is a **fixed corruption process** (i.e. no training involved)
We use Bernoulli diffusion:

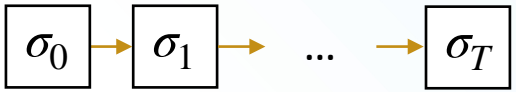
$$f_t(\sigma_t \mid \sigma_{t-1}) = \prod_i \begin{cases} 1 - p_t, & \sigma_{t,i} = \sigma_{t-1,i} \\ p_t, & \sigma_{t,i} = -\sigma_{t-1,i} \end{cases}$$

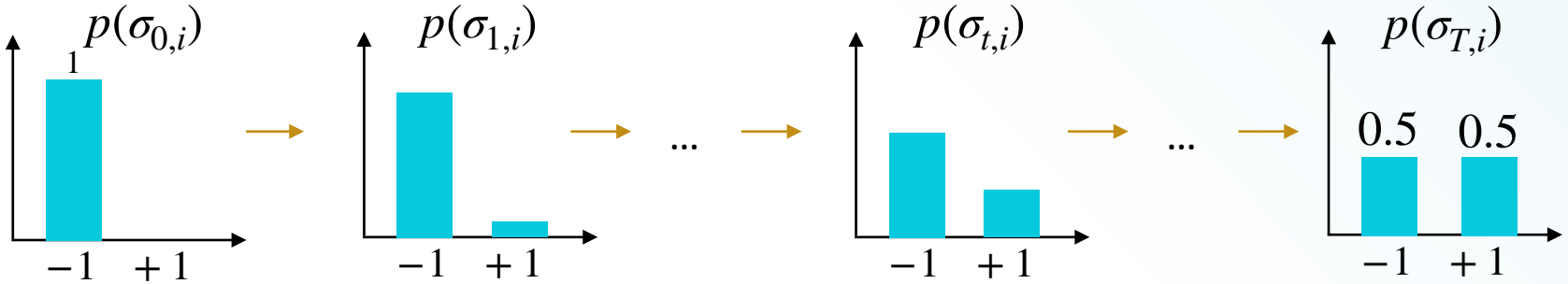
Keep the same spin

Flip the spin

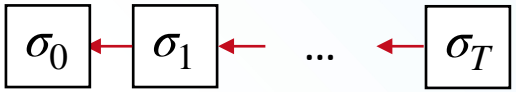
Data-Free Training: Part 1 (Discrete Diffusion)







Data-Free Training: Part 1 (Discrete Diffusion)



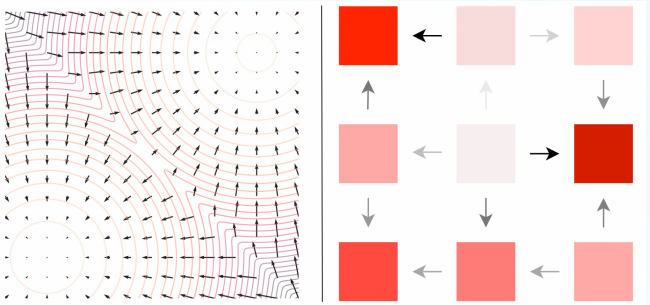
Reverse process is a **learned denoising process.**

Concrete score estimator:

$$\nabla_\sigma \log p(\sigma) = \left[\frac{p(\hat{\sigma})}{p(\sigma)}\right]_{\hat{\sigma}} - 1$$

$\hat{\sigma}$ is a neighboring state in state space, not neighboring sites

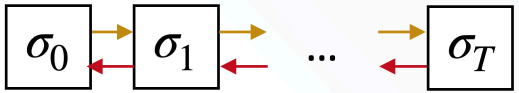
$s_\theta(\hat{\sigma}, \sigma)$



Stein scores
(Continuous diffusion)

Concrete scores
(Discrete diffusion)

Data-Free Training: Part 1 (Discrete Diffusion)



Path: $\Gamma = (\sigma_0, \sigma_1, \ldots \sigma_T)$

Target Boltzmann Path Distribution:

$$\tilde{P}_{\beta,J}(\Gamma) = e^{-\beta H_J(\sigma_0)} \prod_{t=1}^{T} f_t(\sigma_t \mid \sigma_{t-1})$$

Learned Denoising Path Distribution:

$$Q_\theta(\Gamma \mid J) = p_T(\sigma_T) \prod_{t=1}^{T} q_\theta(\sigma_{t-1} \mid \sigma_t, J, t)$$

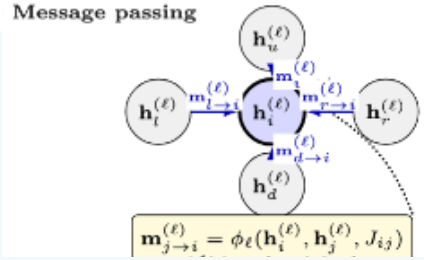
Data-Free Training: Part 2 (Log Variance Loss)

$$\min_{\theta} \; \mathbb{E}_J \left[\mathrm{Var}_{\Gamma|J\sim Y} \left(\log Q_\theta(\Gamma) - \log \tilde{P}_{\beta,J}(\Gamma) \right) \right]$$

Log Variance Loss
[Richter & Berner: arXiv:2307.01198]

- Gradient variance is reduced (due to baseline centering trick)
- Gradient computation is efficient (due to detached samples and simple evaluation function)

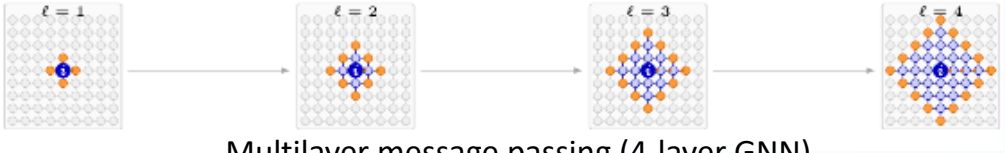
GNN Architecture



Node neural network $\mathbf{h}_i^{(0)} = f_{NN}(\sigma_{i,t}, t)$

Message passing neural network $\mathbf{m}_{j\to i}^{(\ell)} = \phi_\ell(\mathbf{h}_i^{(\ell)}, \mathbf{h}_j^{(\ell)}, J_{ij})$

J_{ij} explicitly models the edge interactions



Multilayer message passing (4-layer GNN)

GNN performs better than CNN

Other architectures

Transformer VAN (tVAN) [Białas+:arXiv:2604.27738]

(15:45 Wednesday: Talk by Dawid Zapolski)

Pathwise Importance Weighting

Pathwise IW:

$$W(\Gamma) = \frac{\tilde{P}_{\beta,J}(\Gamma)}{Q_\theta(\Gamma \mid J)}$$

Endpoint IW:

$$w(\sigma_0) = \frac{\tilde{p}(\sigma_0)}{q_\theta(\sigma_0)}$$

$q_\theta(\sigma_0)$ is not available in diffusion, it is only available in AR. Discrete diffusion is not bijective

Drawback: ESS is low

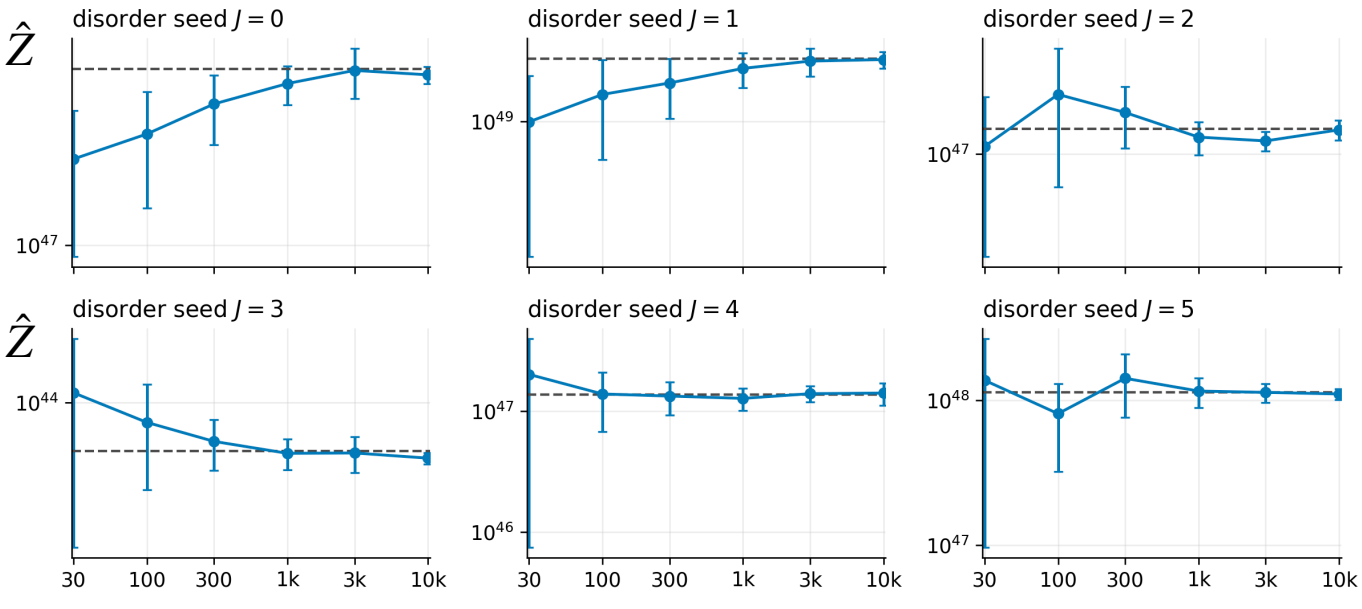
Disorder-Averaged Observable Estimation:

1. Generate K paths for every $n_J \in [1, \ldots, N_J]$
2. Estimate $\langle O \rangle$
3. $\left[\langle O \rangle\right] = \mathbb{E}_J \left[\langle O \rangle\right]$

Experiment Results

$L = 8,\ \beta = 1.2$

Estimating Partition Function $\hat{Z}$ on Unseen Disorders J



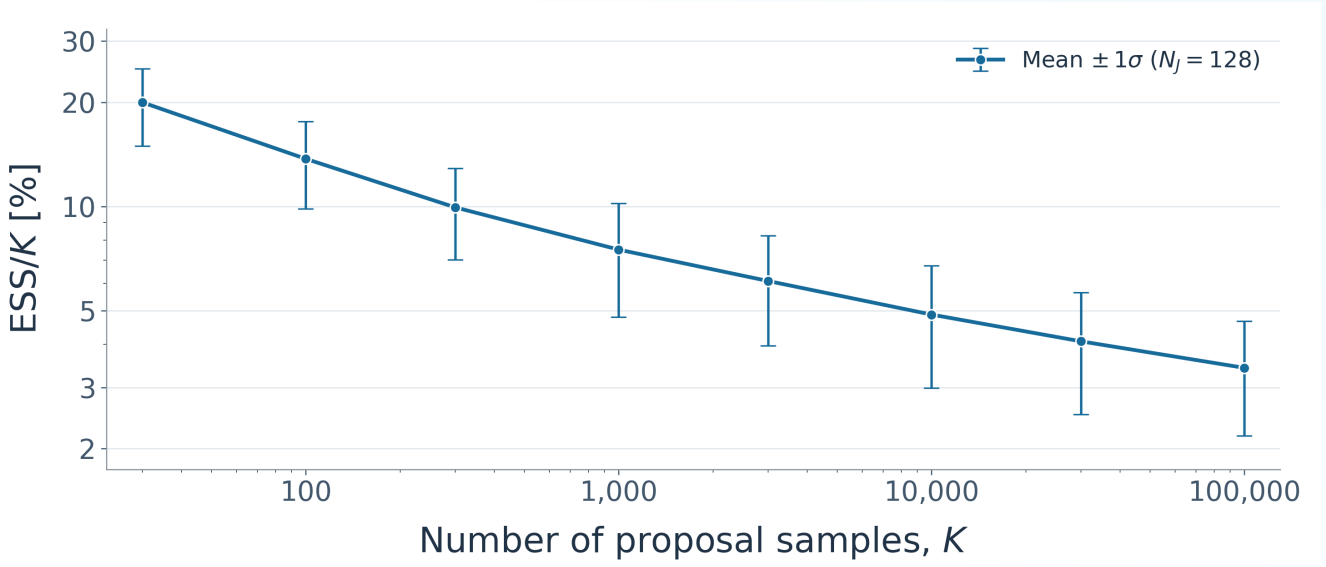
Partition Function

$$\hat{Z}_\beta = \mathbb{E}_{Q_\theta}[W] \approx \frac{1}{K} \sum W_k$$

Groundtruth Z

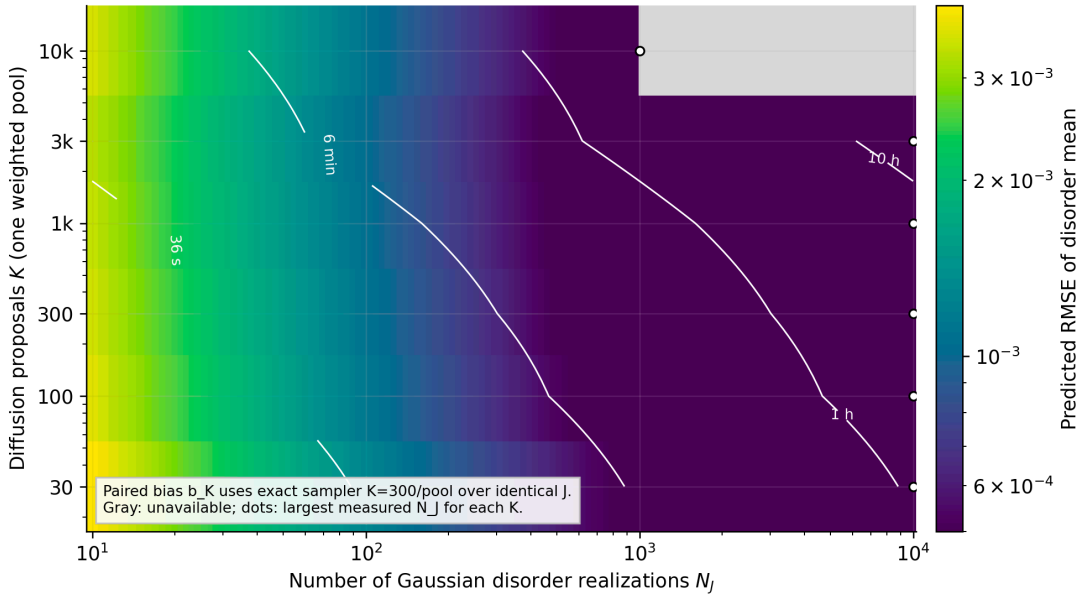
Discrete Diffusion
+ Importance Weighting

Pathwise ESS



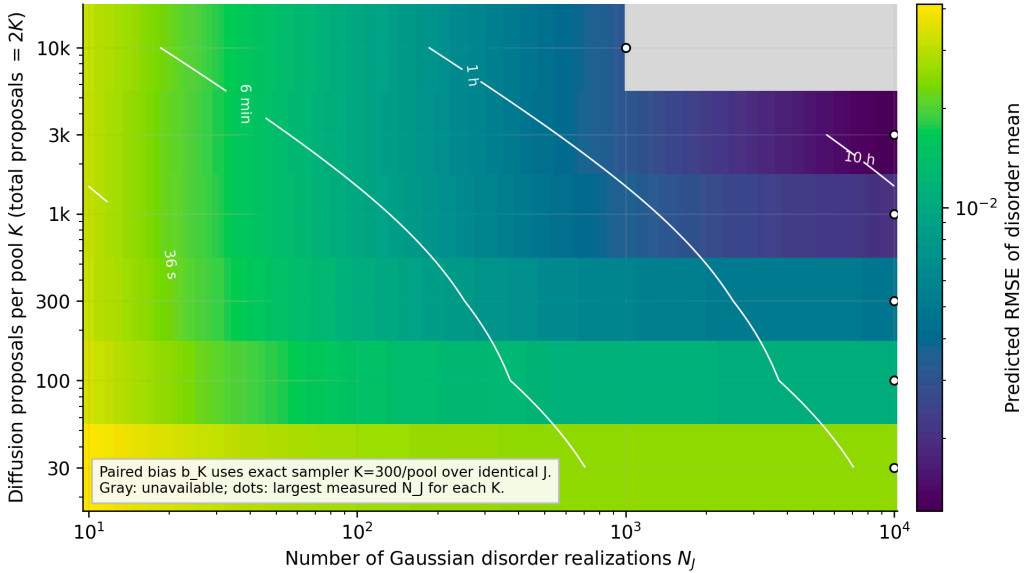
Magnetization $m(\sigma) = \frac{1}{N} \sum_i \sigma_i$

$$\widehat{\left[\langle m^2 \rangle\right]} = \mathbb{E}_J \left[\frac{\sum_k W_{J,k}\, m(\sigma_{t=0})^2}{\sum_k W_{J,k}} \right] \approx \frac{1}{N_J} \sum_{n_J} \left(\frac{\sum_k W_{n_J,k}\, m(\sigma_{t=0})^2}{\sum_k W_{n_J,k}} \right)$$

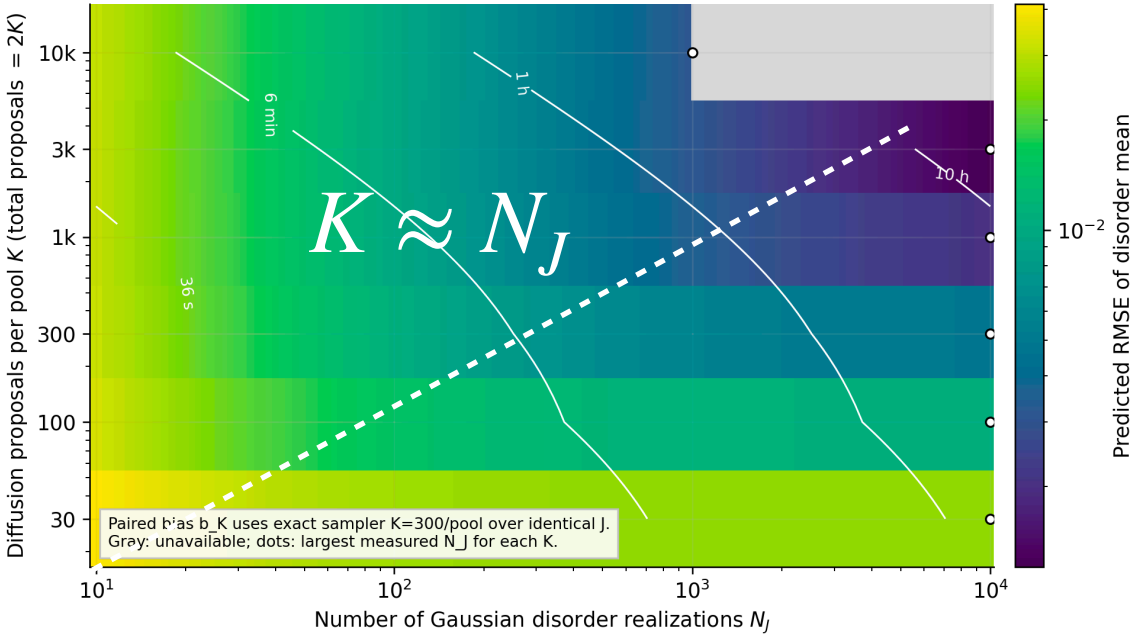


Overlap $q(\sigma, \sigma') = \frac{1}{N} \sum_i \sigma_i \sigma'_i$

$$\widehat{[\langle q^2 \rangle]} = \mathbb{E}_J \left[\frac{\sum_{k,l} W_{J,k} W'_{J,l}\, q(\sigma_{J,k,t=0}, \sigma'_{J,l,t=0})^2}{\left(\sum_k W_{J,k}\right)\left(\sum_l W'_{J,l}\right)} \right] \approx \frac{1}{N_J} \sum_{n_J} \left(\frac{\sum_{k,l} W_{n_J,k} W'_{n_J,l}\, q(\sigma_{n_J,k,t=0}, \sigma'_{n_J,l,t=0})^2}{\left(\sum_k W_{n_J,k}\right)\left(\sum_l W'_{n_J,l}\right)} \right)$$



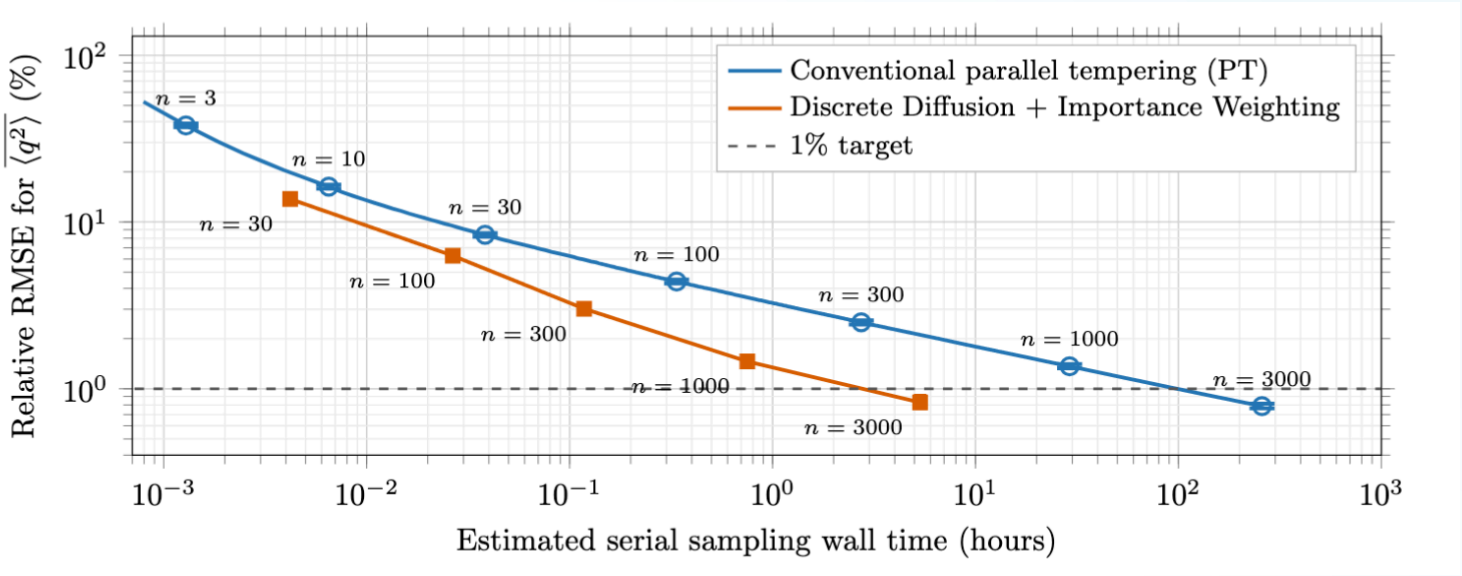
Set optimal $K^\star = N_J$



Computational Comparison vs Conventional PT

We set $n = K = N_J$

$$L = 8$$



Conclusion

- Conditional diffusion sampler, conditioned on J's, for efficient disorder averaging.
- Data-free training without pre-generated equilibrium data (collecting data is costly)
- Demonstrated the performance on $L = 8$, limitations at larger lattices

BIFOLD

Thank you for your attention!

www.bifold.berlin

DSM vs TSM

$$L^{\mathrm{DSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{Uniform}(t)} \left[\| \boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) \|^2 \right].$$

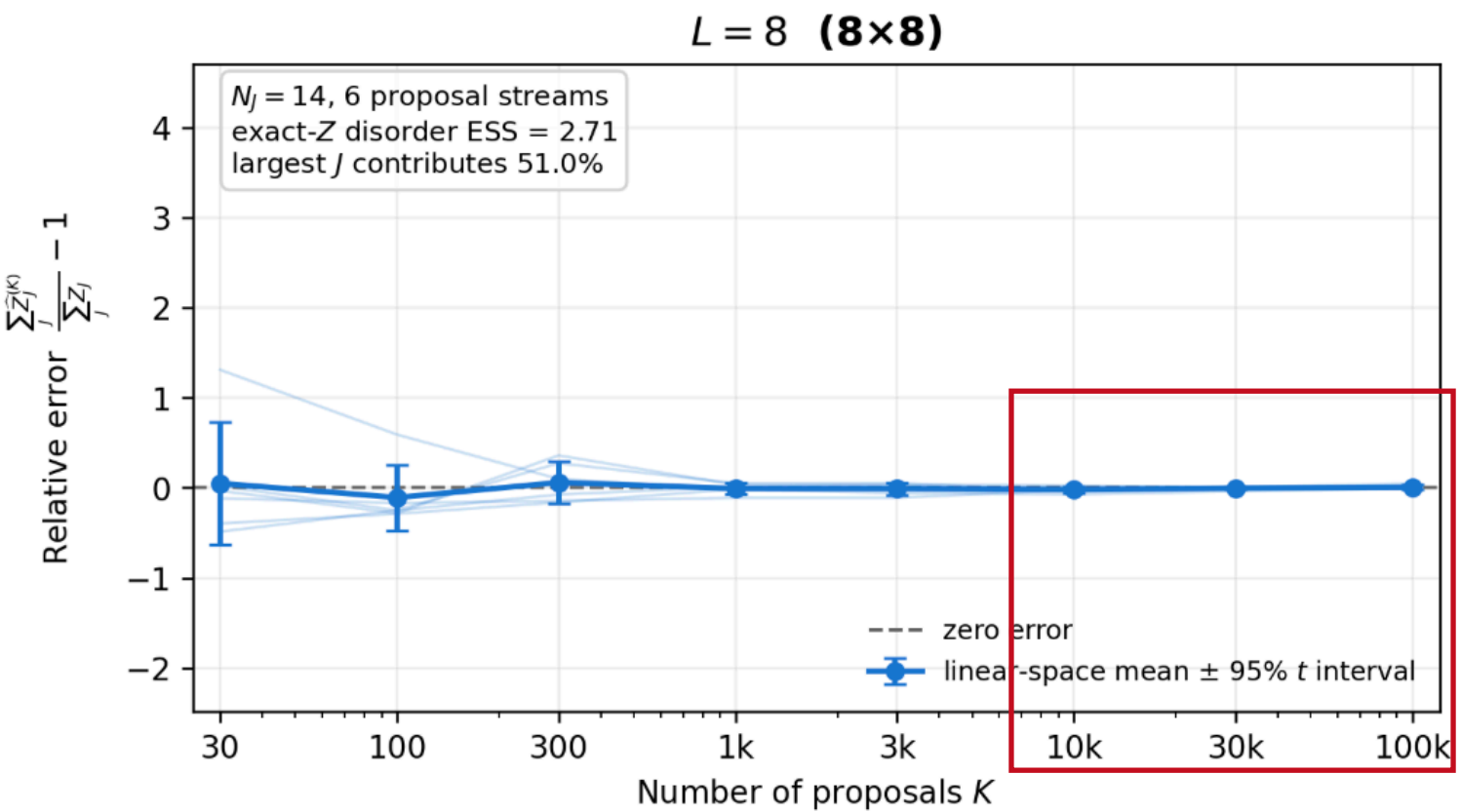
Only the information source:
$\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$ required

Gaussain kernel

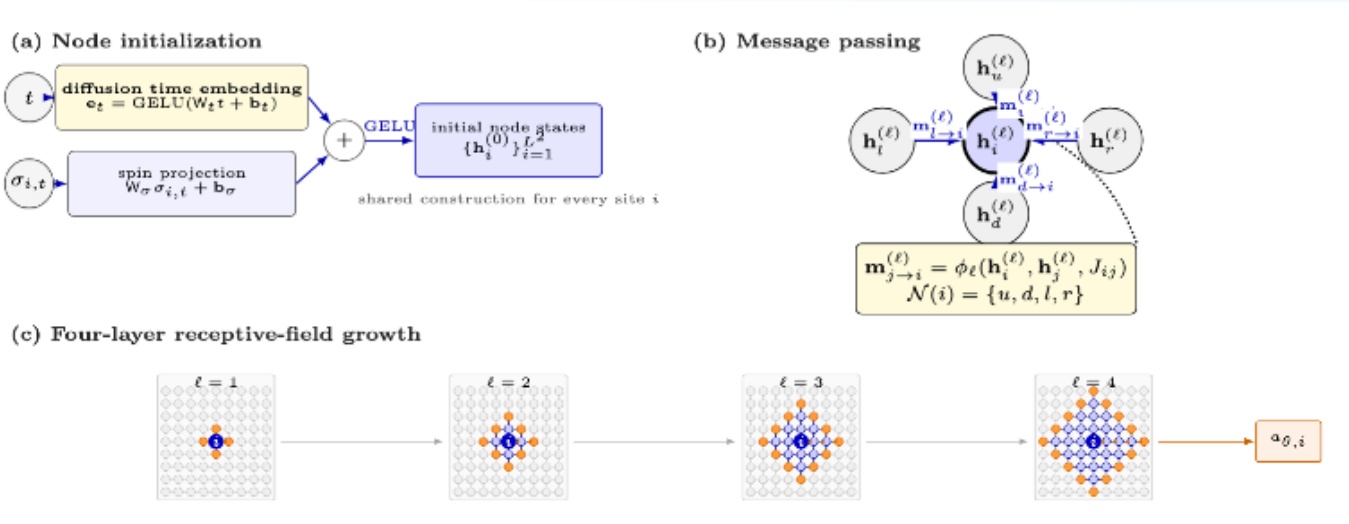
$$L^{\mathrm{TSM}} = \mathbb{E}_{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{Uniform}(t)} \left[\| \boldsymbol{s}_\theta(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_0} \log p_0(\boldsymbol{x}_0) \|^2 \right].$$

Sufficient if the support is covered:
Network learns TSI target as a fixed point.

True score at t=0


$L = 8$ **(8×8)**
$N_J = 14$, 6 proposal streams
exact-Z disorder ESS = 2.71
largest J contributes 51.0%
Relative error $\frac{\sum_J \bar{Z}_J^{(K)}}{\sum_J Z_J} - 1$
4
3
2
1
0
−1
−2
30
100
300
1k
3k
10k
30k
100k
Number of proposals K
zero error
linear-space mean ± 95% t interval

GNN Architecture



Concrete score → reverse rates → denoising transition

Concrete score estimator:

$$s_{\theta,i}(\boldsymbol{\sigma}, t, J) \approx \frac{p_t(\boldsymbol{\sigma}^{(i)} \mid J)}{p_t(\boldsymbol{\sigma} \mid J)} - 1$$

Rate

$$b_{\theta,i}(\boldsymbol{\sigma}, t, J) = p_t \left[1 + s_{\theta,i}(\boldsymbol{\sigma}, t, J)\right]$$

Denoising transition

$$q_\theta(\boldsymbol{\sigma} \mid \boldsymbol{\sigma}, t, J) = 1 - \Delta t \sum_i b_{\theta,i}(\boldsymbol{\sigma}, t, J) + O(\Delta t^2)$$

$$q_\theta(\boldsymbol{\sigma}^{(i)} \mid \boldsymbol{\sigma}, t, J) = \Delta t \, b_{\theta,i}(\boldsymbol{\sigma}, t, J) + O(\Delta t^2)$$

Concrete score → reverse rates → denoising transition

Generalization of gradients: Finite differences

$$\nabla_{\mathbf{x}} p(\mathbf{x}) = \left[p(\hat{\mathbf{x}}) - p(\mathbf{x}) \right]_{\hat{\mathbf{x}}} \quad \text{where } \hat{\mathbf{x}} \text{ is a neighbor of } \mathbf{x}$$

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \left[\frac{p(\hat{\mathbf{x}})}{p(\mathbf{x})} \right]_{\hat{\mathbf{x}}} - 1$$

Concrete scores

Data-Free Training: Part 2 (Log Variance Loss)

$$\min_{\theta} \mathbb{E}_J \left[\mathrm{Var}_{\Gamma|J\sim Y} \left(\log Q_\theta(\Gamma) - \log \tilde{P}_{\beta,J}(\Gamma) \right) \right]$$

[Richter & Berner: arXiv:2307.01198]

- Gradient variance is reduced (due to baseline centering)
- Gradient computation is efficient (due to detached generated samples from the computation graph and simply evaluate this log ratio.
- (Remove Avoids differentiation through discrete sampling
  - With rKL: $L_{\mathrm{rKL}}(\theta) = \mathbb{E}_{\Gamma\sim Q_\theta} \left[R_\theta(\Gamma) \right]$.
  - With LV: $L_{rKL}(\theta) = \mathbb{E}_{\Gamma\sim Q_\theta} \left[\log Q_\theta(\Gamma) - \log \tilde{P}_{\beta,J}(\Gamma) \right]$

    $\Gamma \sim Y$ can be sampled from any reference process Y. In practice, we can choose $Y = Q_{stopgrad(\theta)}$ but it does not guarantee protection from mode collapse. (Exploration vs Exploitation tradeoff))