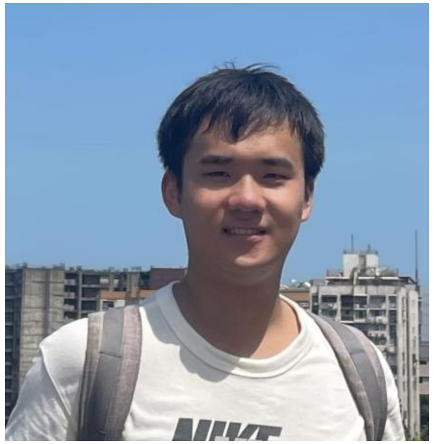
Neural Flow Shortcut Samplers

Yingzhen Li

yingzhen.li@imperial.ac.uk

Based on preliminary work:
Chen*, Ou* and Li,
“Neural Flow Samplers with Shortcut Models”
https://arxiv.org/abs/2502.07337

Wuhao Chen



Zijing Ou

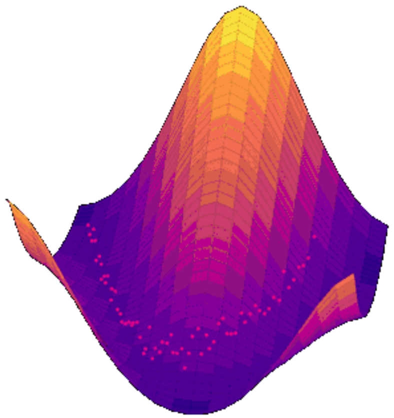
Sampling from Energy Density

Task: sample from $\pi(x) := \frac{1}{Z} \exp[-E(x)]$

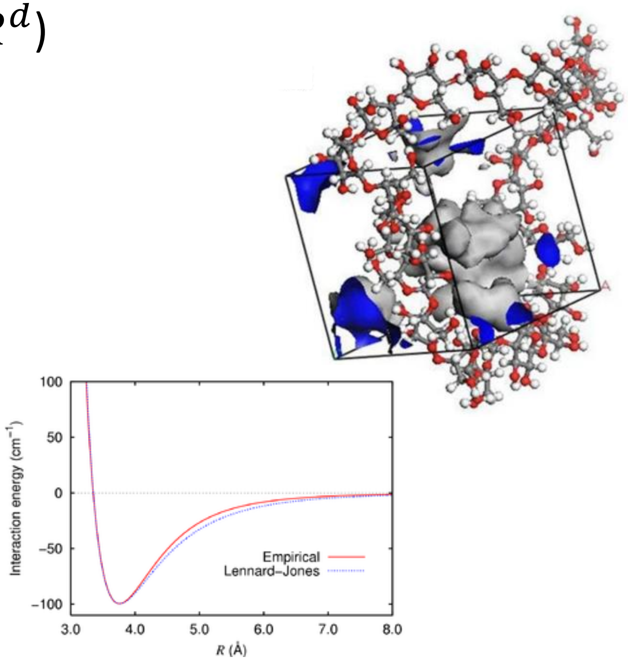
$$Z := \int \exp[-E(x)]\, dx \qquad (x \in R^d)$$

$$P(\theta | D) = \frac{P(\theta)P(D \mid \theta)}{P(D)}$$

Bayesian Inference



Energy-based Models



Molecular Dynamics Simulation

MCMC, SMC & Transport

Task: sample from $\pi(x) := \frac{1}{Z}\exp[-E(x)]$

MCMC

- Find a transition kernel invariant to $\pi(x)$
- Run MCMC transitions until ”convergence”

This work (training)

SMC

- Based on Importance Sampling
- Define a sequence of proposal distributions towards $\pi(x)$
- Reweighting & Resampling

This work (training)

Transport

- Define an initial distribution $p_0(x)$
- Find a transport map T such that

$$x_1 \sim \pi(x)$$
$$\Leftrightarrow$$
$$x_0 \sim p_0(x), x_1 = T(x_0)$$

This work (sampling)

Continuous Normalising Flows

- Transport via Continuous Normalising Flows (CNFs):

$$x_1 = T(x_0), \qquad T(x_0) := x_0 + \int_0^1 v_t(x_t)dt$$

- Probability density evolves: $\{p_t(x)\}_{t\in[0,1]}$ satisfy

$$\partial_t \log p_t(x) = -\nabla_x \cdot v_t(x) - \langle \nabla_x \log p_t(x), v_t(x) \rangle$$

- Notice the difference from change-of-variable rule:

$$\partial_t \log p_t(x_t) = -\nabla_x \cdot v_t(x_t)$$

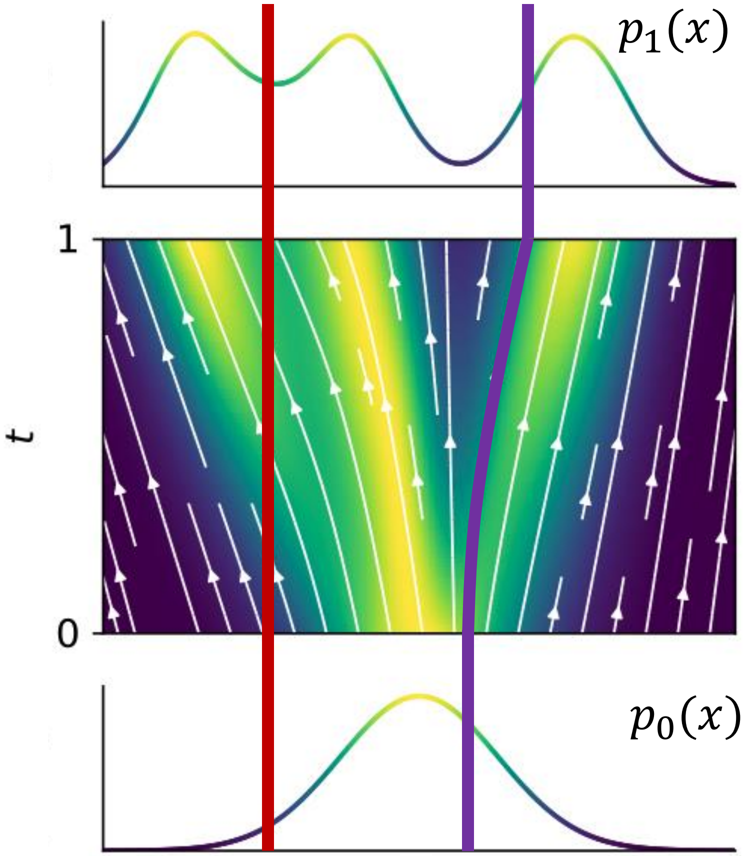


Figure adapted from Grathwohl et al. ICLR 2019

$\pi(x) = \frac{1}{Z}\exp[-E(x)]$

Neural Flow Sampler

E.g., tempering:
$E_t(x) = \beta_t E_0(x) + (1-\beta_t)E(x),$
$\beta_0 = 1, \beta_1 = 0$

- Specify a density path:

$$\{p_t(x)\}_{t\in[0,1]}: \qquad p_t(x) = \frac{1}{Z_t}\exp[-E_t(x)],$$

$$p_0(x) \text{ easy to sample}, \qquad p_1(x) = \pi(x), \text{i.e., } E_1(x) \coloneqq E(x)$$

- Learn a flow model $v_\theta(x,t)$ by minimising L2 error:

$$L(v_\theta) \coloneqq E_{q_t(x)}[|| \, \partial_t \log p_t(x) + \nabla_x \cdot v_\theta(x,t) + \langle \nabla_x \log p_t(x) \, , v_\theta(x,t) \rangle \, ||_2^2]$$

Ensuring continuity equation to hold for every $x \sim q_t(x)$

- Simulate samples from $\pi(x)$ (approximately) by solving ODE:

$$x_0 \sim p_0(x), \qquad x_1 \coloneqq x_0 + \int_0^1 v_\theta(x_t,t)dt$$

Tian et al. Liouville Flow Importance Sampler. ICML 2024
Mate and Fleuret. Learning Interpolations between Boltzmann Densitie. TMLR 2023

$\pi(x) = \frac{1}{Z}\exp[-E(x)]$

Neural Flow Sampler

$= -\nabla_x E_t(x)$

Training: $L(v_\theta) := E_{q_t(x)}[\| \partial_t \log p_t(x) + \nabla_x \cdot v_\theta(x,t) + \langle \nabla_x \log p_t(x), v_\theta(x,t)\rangle \|_2^2]$

Sampling: $x_0 \sim p_0(x), \qquad x_1 := x_0 + \int_0^1 v_\theta(x_t, t)dt$

- Challenges:
  - Selecting the "training data" distribution $q_t(x)$ and estimating the expectation
    - Not necessary for $q_t(x) = p_t(x)$ but ideally $q_t(x) \approx p_t(x)$
  - Estimating $\partial_t \log p_t(x)$:

$$p_t(x) := \frac{1}{Z_t}\exp[-E_t(x)] \quad \Rightarrow \quad \partial_t \log p_t(x) = -\partial_t E_t(x) - \partial_t \log Z_t$$

(intractable)

  - Solving the ODE flow simulation in a fast way

$\pi(x) = \frac{1}{Z}\exp[-E(x)]$

Using "Training Data" $q_t(x)$

$$L(\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \nabla_x \cdot v_\theta(x,t) + \langle \nabla_x \log p_t(x), v_\theta(x,t) \rangle \|_2^2]$$

- Estimating expectation under $q_t(x) \approx p_t(x)$ via *velocity-driven* SMC:
  - A typical SMC method (e.g., Hamiltonian AIS):
    - Pick $0 = t_0 < t_1 < t_2 < \cdots < t_M = 1$ and run SMC with path $\{p_{t_m}(x)\}_{m=0}^{M}$ as proposals
    - Compute the importance weights by accumulating density ratios through time
    - Resampling is required by monitoring ESS
  - The steps for approximately drawing samples from $p_{t_m}(x)$:
    - Transport from previous step: $\tilde{x}_{t_m} = x_{t_{m-1}} + \int_{t_{m-1}}^{t_m} v_\theta(x_t, t)dt$ ("prediction")
    - Run (short-chain) HMC: $x_{t_m} = HMC(\tilde{x}_{t_m})$ ("correction")

Neal. Annealed Importance Sampling. Stats. Comp., 2001
Neal. MCMC using Hamiltonian Dynamics. Handbook of MCMC, 2010

$\pi(x) = \frac{1}{Z}\exp[-E(x)]$

Estimating $\partial_t \log Z_t$

Monte Carlo estimate can have high variance!

$$\partial_t \log Z_t = \frac{1}{Z_t}\partial_t \int \exp[-E_t(x)]\, dx = -\frac{1}{Z_t}\int \exp[-E_t(x)]\, \partial_t E_t(x) dx = -E_{p_t(x)}[\partial_t E_t(x)]$$

- Solution: Stein control variate with $x^k \sim p_t(x)$

(Langevin-Stein Operator)

$$-E_{p_t(x)}[\partial_t E(x)] \approx \frac{1}{K}\sum_{k=1}^{K} -\partial_t E_t(x^k) + \beta[\nabla_x \cdot v_\theta(x^k, t) + \langle \nabla_x \log p_t(x^k), v_\theta(x^k, t)\rangle]$$

- Stein's Identity ensures unbiasedness: for any $v_\theta(x, t)$

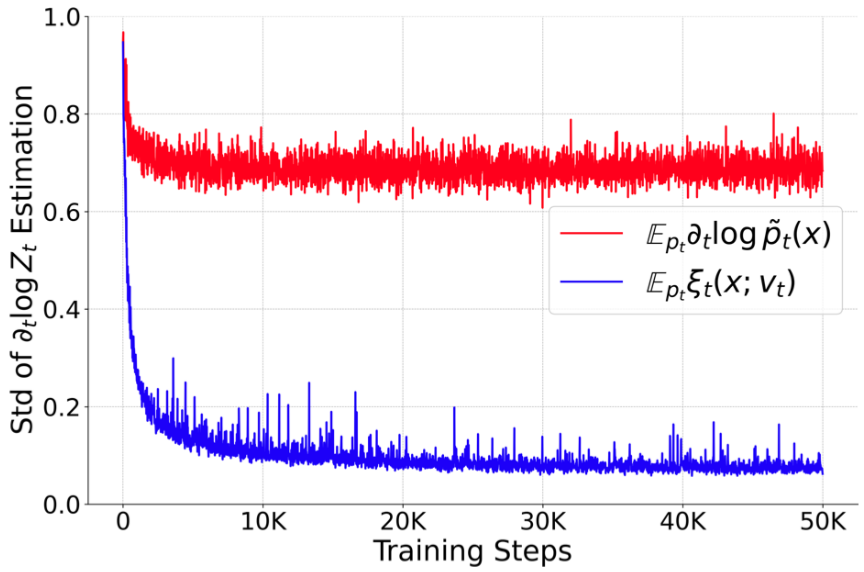
$$E_{p_t(x)}[\nabla_x \cdot v_\theta(x,t) + \langle \nabla_x \log p_t(x), v_\theta(x,t)\rangle] = 0$$

- Continuity equation for *globally optimal* $v_\theta(x, t)$

$$-\partial_t E_t(x) + [\nabla_x \cdot v_\theta(x,t) + \langle \nabla_x \log p_t(x), v_\theta(x,t)\rangle] = \partial_t \log Z_t$$

$$:= \xi_t(x; v_\theta(x,t))$$

$\Rightarrow$ Variance is 0 when $\beta = 1$ and $v_\theta(x, t)$ is optimal



Liu et al. Action-dependent Control Variates for Policy Optimization via Stein Identity. ICLR 2018

Speeding up with Shortcuts

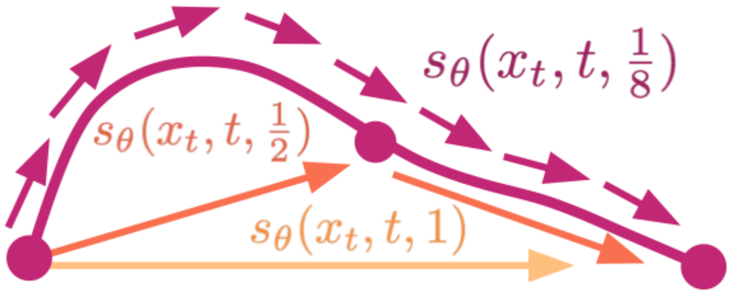
Sampling: $x_0 \sim p_0(x), \quad x_1 \coloneqq x_0 + \int_0^1 v_\theta(x_t, t)dt$

Naïve Euler method requires a lot of steps!

Solution: Shortcut models

$$x_{t+d} \leftarrow x_t + s_\theta(x_t, t, d)d$$

- A shortcut model s_θ is a valid ODE solver if for any x_t:
  - $s_\theta(x_t, t, 0) \coloneqq v_\theta(x_t, t)$
  - $s_\theta(x_t, t, 2d) = \frac{1}{2}s_\theta(x_t, t, d) + \frac{1}{2}s_\theta(x_{t+d}, t+d, d)$



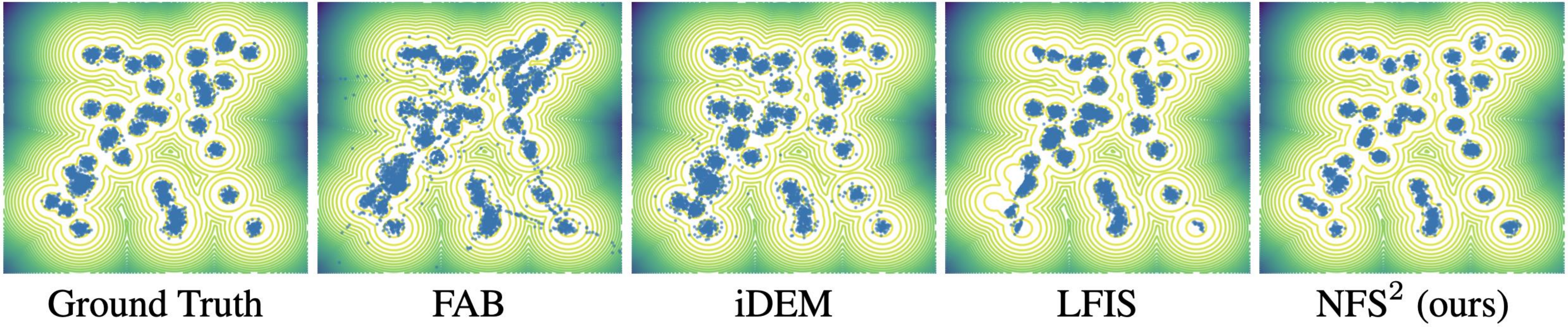
- Training a neural flow shortcut sampler:

$$\tilde{L}(s_\theta) \coloneqq L\big(s_\theta(\cdot,\cdot,0)\big) + E_{q(d)}\left[\|s_\theta(x_t, t, 2d) - \tfrac{1}{2}s_\theta(x_t, t, d) - \tfrac{1}{2}s_\theta(x_{t+d}, t+d, d)\|_2^2\right]$$

flow learning — enforcing consistency

Frans et al. One Step Diffusion via Shortcut Models. ICLR 2025

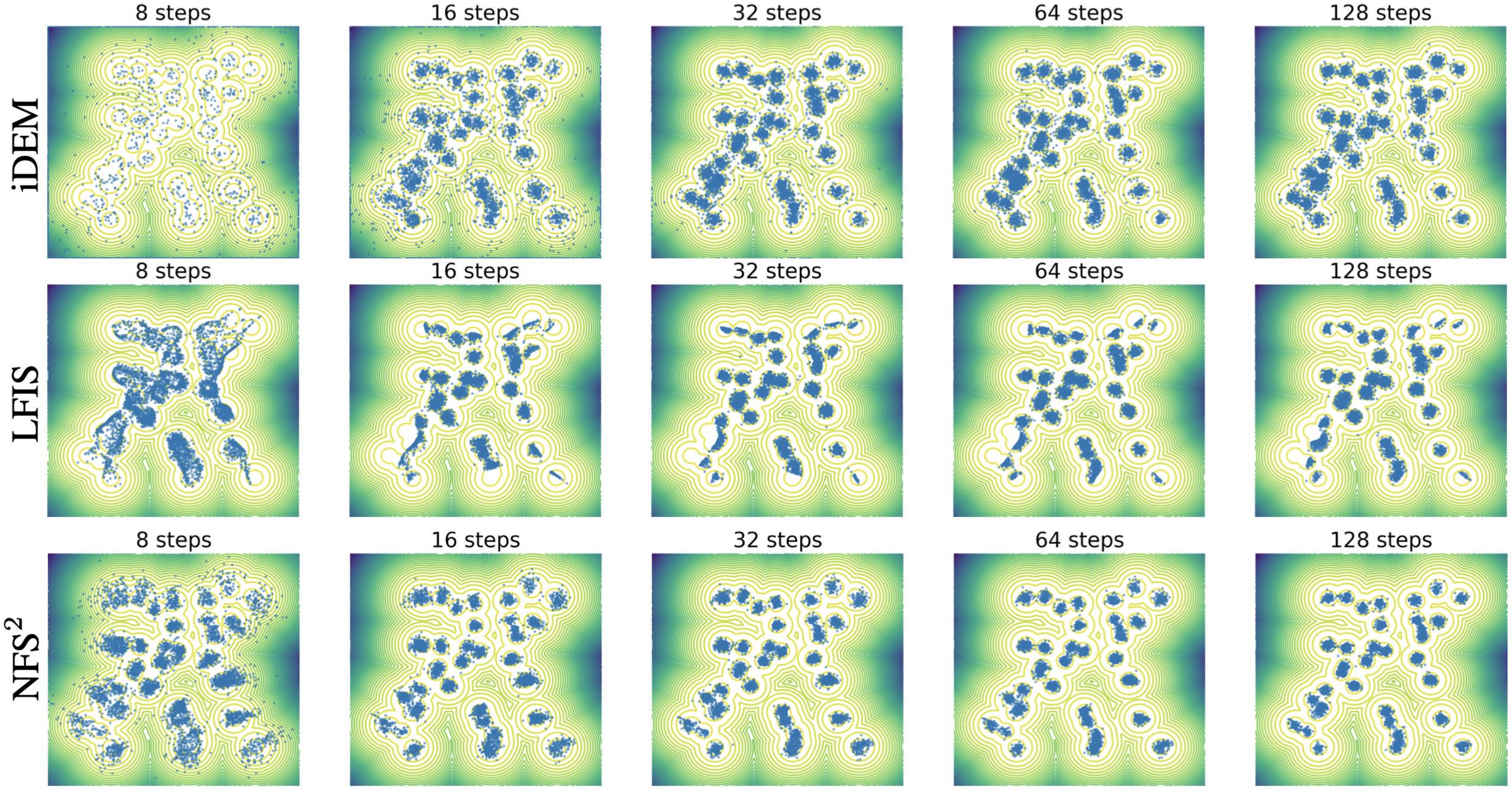
Example: Mixture of 40 Gaussians



- "Ground Truth": samples from mixture of Gaussians
- FAB: normalising flow transport map, trained by alpha-divergence, "data" from AIS + replay buffer
- iDEM: diffusion-based, score estimation via importance sampling + replay buffer
- LFIS: continuity equation-based loss, no amortization across t, simple importance sampling for $\partial_t \log Z_t$

Midgley et al. Flow Annealed Importance Sampling Bootstrap. ICLR 2023
Akhound-Sadegh et al. Iterated Denoising Energy Matching for Sampling from Boltzmann Densities. ICML 2024
Tian et al. Liouville Flow Importance Sampler. ICML 2024

Example: Mixture of 40 Gaussians

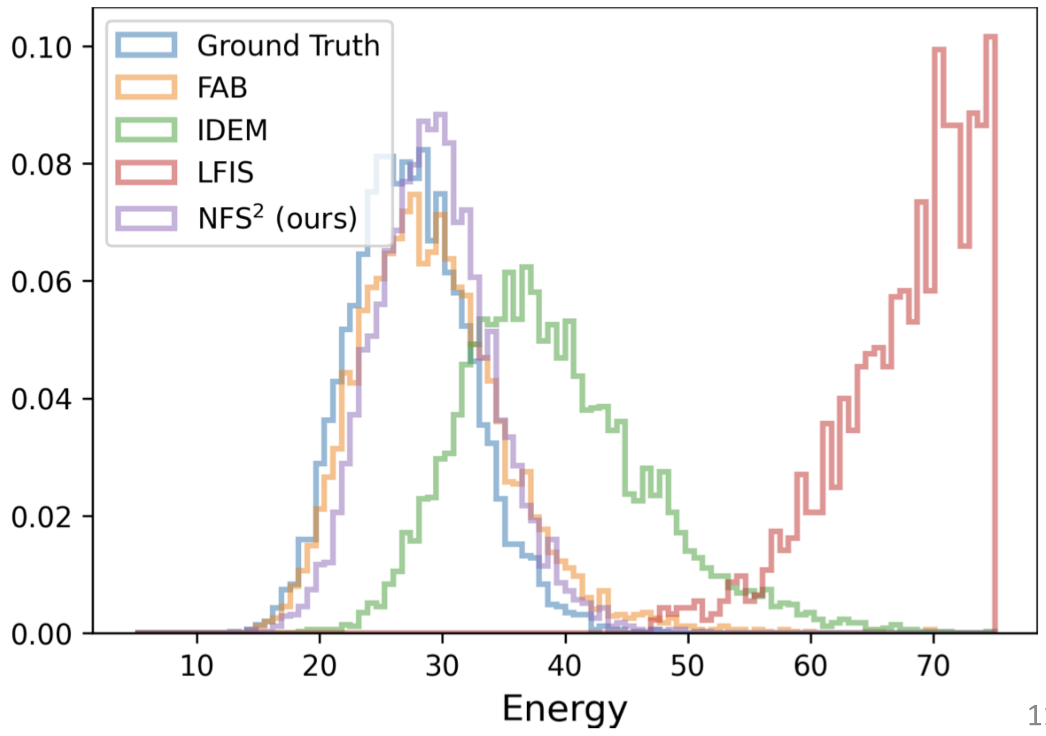


Example: Many-Well 32-Dim

$$\pi(x) = \prod_{i=1}^{16} \pi(x_{2i-1}, x_{2i}),$$

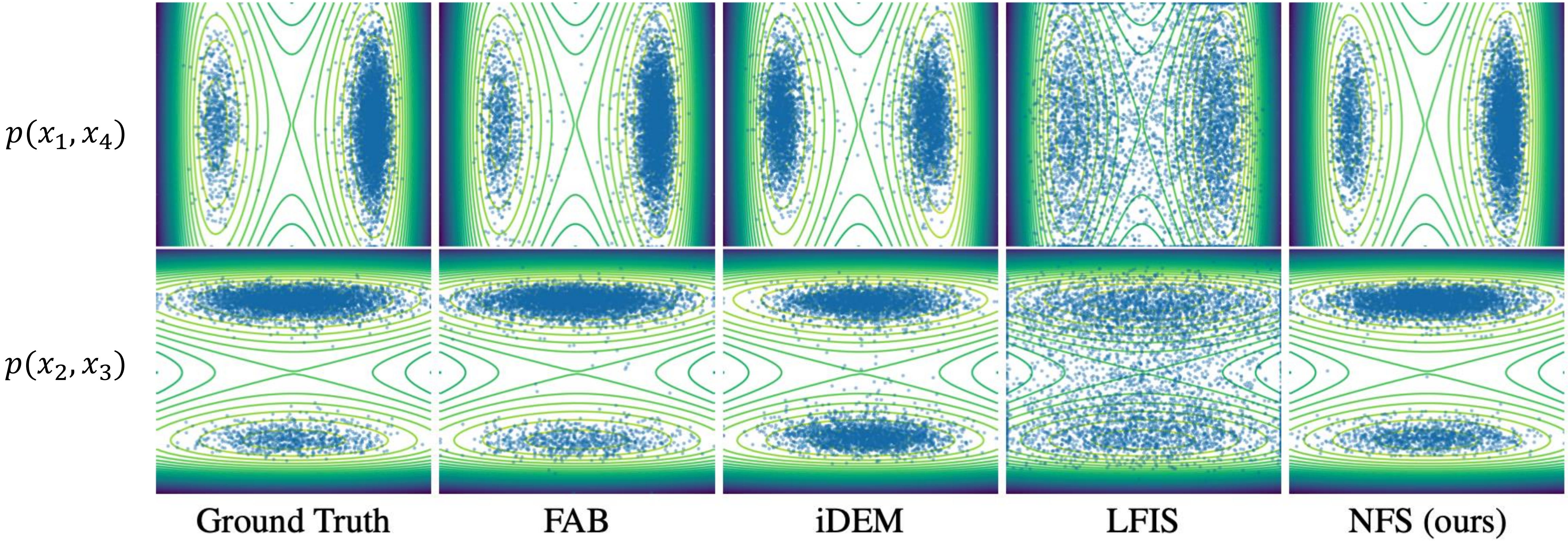
$$\log \pi(x_{2i-1}, x_{2i}) = -x_{2i-1}^4 + 6x_{2i-1}^2 + 0.5\, x_{2i-1} - 0.5x_{2i}^2 + C$$

- $\dim(x) = 32$
- $2^{16} = 65{,}536$ symmetric modes
- “Ground Truth” samples generated by sampling from the marginals
    - Rejection sampling for x_{2i-1}
    - Gaussian for x_{2i}



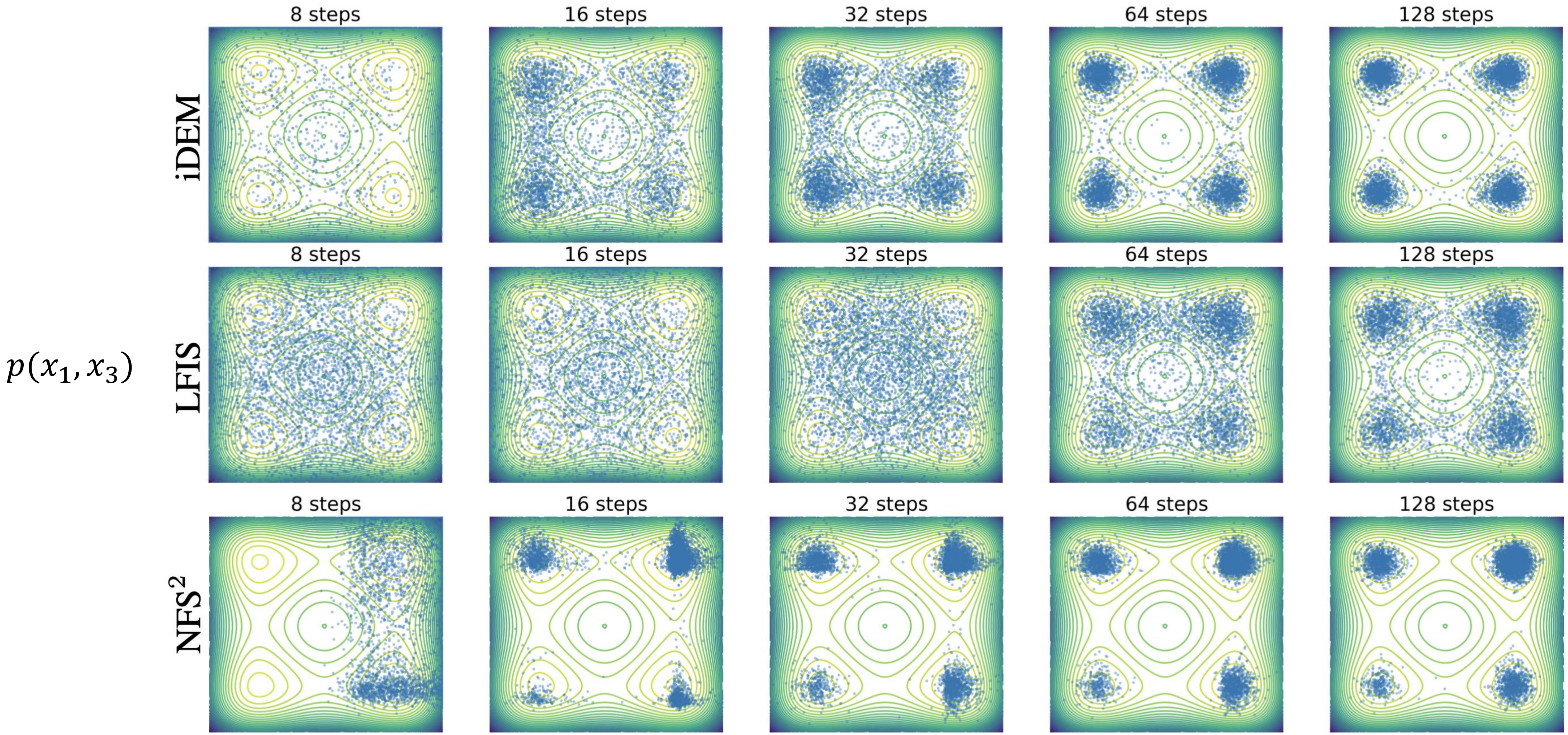
Midgley et al. Flow Annealed Importance Sampling Bootstrap. ICLR 2023

Example: Many-Well 32-Dim



- “Ground Truth”: rejection sampling for odd dimensions + Gaussian sampling for even dimensions
- FAB: normalising flow transport map, trained by alpha-divergence, “data” from AIS + replay buffer
- iDEM: diffusion-based, score estimation via importance sampling + replay buffer
- LFIS: continuity equation-based loss, no amortization across t, simple importance sampling for $\partial_t \log Z_t$

Example: Many-Well 32-Dim



Quantitative Results

Table 1: Comparison of neural samplers on GMM-40, MW-32, and DW-4 energy functions, with mean and standard deviation based on five evaluations using different random seeds.

Energy →	GMM-40 ($d = 2$)		MW-32 ($d = 32$)		DW-4 ($d = 8$)		
Method ↓	$\mathcal{E}$-$\mathcal{W}_2$	$\mathcal{X}$-TV	$\mathcal{E}$-TV	$\mathcal{X}$-$\mathcal{W}_2$	$\mathcal{E}$-$\mathcal{W}_2$	$\mathcal{E}$-TV	$\mathcal{D}$-TV
FAB (Midgley et al., 2023)	8.89±2.20	0.84±0.19	0.25±0.01	5.78±0.02	0.64±0.20	0.22±0.01	0.09±0.01
iDEM (Sadegh et al., 2024)	1.27±0.21	0.83±0.01	0.63±0.15	8.18±0.04	0.19±0.05	0.21±0.01	0.10±0.01
LFIS (Tian et al., 2024)	0.27±0.21	0.84±0.01	∞	8.89±0.03	6.06±1.05	0.66±0.02	0.29±0.01
NFS^2-128 (ours)	0.46±0.14	0.67±0.00	0.16±0.00	6.17±0.01	0.44±0.03	0.10±0.01	0.07±0.01
NFS^2-64 (ours)	1.32±0.29	0.69±0.01	0.18±0.00	6.34±0.01	0.98±0.16	0.13±0.01	0.11±0.01
NFS^2-32 (ours)	4.38±1.14	0.72±0.01	0.49±0.01	9.05±0.01	14.97±0.82	0.41±0.01	0.28±0.01

DW-4: $E(x) = \frac{1}{2}\sum_{i,j}\left[-4\left(d_{ij} - d_0\right)^2 + 0.9\left(d_{ij} - d_0\right)^4\right], d_{ij} = \left\|x_i - x_j\right\|_2$

- Compared with "ground-truth" samples
- Comparing x-space sample distribution (e.g., x-TV) & energy histogram (e.g., ϵ-TV)
- For DW-4, x-space metric is replaced by d-space metric ("distance between atoms")

Summary of the Recipe

Task: sample from $\pi(x) := \frac{1}{Z}\exp[-E(x)]$

- Idea of Neural Flow Shortcut Sampler in a nutshell:
  - Specify a density path $\{p_t(x)\}_{t\in[0,1]}$ with:
    - Easy-to-sample $p_0(x)$
    - Tractable energy function $p_t(x) \propto \exp[-E_t(x)]$
    - $p_1(x) = \pi(x)$
  - Train a flow sampler to satisfy the continuity equation wrt. $\{p_t(x)\}_{t\in[0,1]}$
    - Selecting a good "training data" distribution (via e.g., SMC)
    - Estimating intractable terms efficiently (with e.g., control variate)
  - In sampling time, generate samples by ODE/flow simulation
    - Shortcut model to speed-up, achieving speed-accuracy trade-off

Chen*, Ou* and Li,
"Neural Flow Samplers with Shortcut Models"
https://arxiv.org/abs/2502.07337

Challenges & Future Work

- Quality of $q_t(x)$ as an approximation to $p_t(x)$
- Computation of divergence $\nabla_x \cdot v_\theta(x, t)$
- Sampling from density with high "energy barrier"
- Adaptive and faster ODE solvers (e.g., adaptive shortcut model?)
- Also optimising the density path $\{p_t(x)\}$?

- Scaling-up the neural samplers to high dimensions?
- Discrete versions of neural flow samplers and shortcuts?
- Simulation-free training?

Holderrieth et al. LEAPS: A discrete neural sampler via locally equivariant networks. arXiv:2502.10843

Appendix: A Coordinate Descent View

- Practical Implementation of Stein control variate with $x^k \sim q_t(x)$ (Langevin-Stein Operator)

$$\partial_t \log Z_t \approx \frac{1}{K}\sum_{k=1}^{K} -\partial_t E_t(x^k) + \beta\left[\nabla_x \cdot v_\theta(x^k, t) + \langle \nabla_x \log p_t\,(x^k), v_\theta(x^k, t)\rangle\right]$$

- Equivalent to performing coordinate descent + Monte Carlo for optimisation w.r.t. v_θ and C_t:

$$L(v_\theta, C_t) \coloneqq E_{q_t(x)}[\| -\partial_t\, E_t(x) - C_t + \nabla_x \cdot v_\theta(x,t) + \langle \nabla_x \log p_t(x)\,, v_\theta(x,t)\rangle \|_2^2]$$

$$\Rightarrow \qquad C_t^* = E_{q_t(x)}[-\partial_t\, E_t(x) + \nabla_x \cdot v_\theta(x,t) + \langle \nabla_x \log p_t(x)\,, v_\theta(x,t)\rangle]$$

With globally optimal $v_\theta(x,t)$, we also have $C_t^* = \partial_t \log Z_t$